Evolution of Efficient Gait with an Autonomous Biped Robot Using Visual Feedback

Krister Wolff **Peter Nordin**

Department of Physical Resource Theory
Chalmers University of Technology
S-412 96 Göteborg, Sweden
E-mail: {wolff, nordin}@fy.chalmers.se

Abstract

In this paper we present the autonomous, walking humanoid robot ELVINA and the first experiments in genetic programming performed with it. A steady state evolutionary strategy is running on the robot's onboard computer. Individuals are evaluated and fitness scores are automatically determined using the robots onboard digital camera and near-infrared range sensor. The experiments are performed in order to optimize a by hand developed locomotion controller. By using this system, we evolved gait patterns that locomote the robot in a straighter path and in a more robust way, than the previously manually developed gait did.

1 INTRODUCTION

The applications of robots with human-like dimensions and motion capabilities -*humanoid robots* are plentiful. Humanoid robots constitute both one of the largest potentials and one of the largest challenges in the fields of autonomous agents and intelligent robotic control. In a world where man is the standard for almost all interactions, humanoid robots have a very large potential acting in environments created for humans. Both in industry and in academia walking humanoid robots attracts an accelerating interest [Nordin & Nordahl, 1999]. In 1996, Honda Corporation presented the P2[1] humanoid robot which is a biped robot that can walk like a human, even up and down stairs. A smaller and lighter robot, P3[1], was introduced in 1997 and recently, they presented the humanoid robot ASIMO[1], which is conceived to function in an actual human living environment in the near future. The Sony Corporation announced in November 2000 the development of a small biped walking robot, SDR-3X[2], which is a platform for exploration of new possibilities for entertainment robots.

In traditional robot control programming, an internal model of the system is derived and the inverse kinematics can be calculated. The trajectory for movement between given points in the working area of the robot is then calculated from the inverse kinematics. Even though this still is a very common approach, we propose for several reason the concept of *genetic programming* for control programming of so-called bio-inspired robots [Dittrich et al, 1998] as e.g. a humanoid. The traditional geometric approach to robot control, based on modeling of the robot and derivation of leg trajectories, is computationally expensive and requires fine tuning of several parameters in the equations describing the inverse kinematics [Nolfi & Floreano, 2000]. Conventional industrial robots are designed in such a way that a model can be easily derived, but for the development of bio-inspired robots, this is not a primary design principle. Thus, a model of the system is very hard to derive or to complex so that a model-based calculation of actuator commands requires to much time for reactive tasks [Dittrich et al, 1998]. For a robot that is conceived to operate in an actual human living environment, it is impossible for the programmer to consider all eventualities in advance. The robot is therefore required to have an adaptation mechanism that is able to cope with unexpected situations.

The primary goal for our work presented in this paper is to evolve a gait pattern, using genetic programming and especially evolutionary strategies [Banzhaf et al, 1998]. To do this, one has to choose between two main alternatives: using a real robot for the evolution, or using a simulated robot. Several experiments with simulations, with different approaches, have been reported recently. Anytime learning can make use of evolutionary computation in a learning module for the robot to adapt to changes in the robot's capabilities without the use of

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

[2] http://www.sony.co.jp/en/SonyInfo/News/Press/200011/00-057E2/

internal sensors [Parker & Mills, 1999]. A methodology for developing simulators for evolution of controllers in *minimal simulations* has been proposed and shown to be successful when transferred to a real, physical octopod robot [Jakobi, 1998]. This was also compared with a controller that was evolved with a real octopod robot [Gomi & Ide 1998]. It was found that it matched better the physical constraints of the robot hardware. Using simulation, ball-chasing behavior has been evolved and successfully transferred to a real AIBO[3] quadruped robot dog [Hornby* et al, 2000]. The collisions between the robot and ball had different results in the real world than in the simulated world, however it did not affect ball-chasing performance. When a high degree of accuracy is necessary, it is desirable to be able to evolve with a physical robot. We want to show that evolution of controllers with complex, physical robots can be carried out in reality, although evolving with a simulator would do it many times faster.

The first attempt in using a real, physical robot to evolve gait patterns was made at the University of Southern California [Lewis et al, 1992]. Neural networks were evolved as controllers to get a tripod gait for a hexapod robot with two degrees of freedom for each leg. Recently, a group of researchers at Sony Corporation presented the results of their work with evolving locomotion controllers for dynamic gait of their quadruped robot dog AIBO [Hornby et al, 1999] and [Hornby et al, 2000]. These results show that evolutionary algorithms can be used on complex, physical robots to evolve non-trivial behaviors on those robots. In previous evolution with physical robots has a humanoid, biped robot not been used.

Our test problem is that of developing locomotion controllers for static gaits for our biped robot ELVINA. Evolution of static walking with a biped robot is much more difficult than it is with a robot that has a greater number of legs. A static gait requires that the projection of the center of mass of the robot on the ground lies within the support polygon formed by feet on the ground [Nolfi & Floreano, 2000]. This is obviously easier to fulfill with a robot that got four, six, eight or more legs. However, dealing with biped locomotion leads us into a partly different problem domain. When a biped robot is walking (static), it is supported only by one foot at the ground during an appreciable period of time. Only this single foot then constitutes its support polygon. For a biped robot, the area of the support polygon is relatively small, compared to the altitude of where its center of mass is located. The corresponding measure for a robot that got four or more legs is relatively larger. Therefore it is easier for a robot with many legs to maintain balance than it is for a biped robot, as the motion of walking dynamically changes the stability of the robot.

The rest of this paper is organized as follows. Section 2 consists of a description of our robot ELVINA. Section 3 describes the evolutionary algorithm used for our evolution. In section 4, we describe the setup of our experiment and how the robot's sensors are used. In section 5, we present the results of our experiments. We discuss these results in section 6. Finally, section 7 is a conclusion of this work.

2 ROBOT PLATFORM

The robot used in our experiments is ELVINA, which is a simplified, scaled model of a full-size humanoid with body dimensions that mirrors the dimensions of a human. The ELVINA humanoid is a fully autonomous robot with onboard power supply and computer, but many experiments are performed with external power supply. It is 28cm tall and it weights about 1490g including batteries. Each of the two legs has 5 degrees of freedom, of which 4 DOF is active and 1 DOF is passive. The head, the torso and the arms has 1 DOF each, giving a total of 14 DOF. The robot is equipped with a digital CMOS color camera, mounted in its head. The computer is attached to the back of the robot's body. The body also houses a near-infrared PSD (position sensitive detector) which is used to determine distances to nearby objects. In its present status, the robot is capable of static walking.

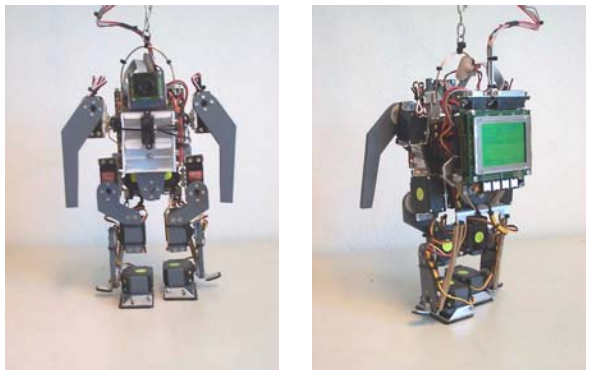

Figure 1. Pictures of the ELVINA Humanoid.

2.1 BODY

The body structure of the robot is constructed with the actuators as the main elements. The actuators that constitute the different sections of the body are connected to each other with parts made of 6mm thick PVC board so that they together form the robot body. See figure 1. This PVC material fulfills all necessary requirements since it is inexpensive and lightweight, yet strong and durable.

2.2 ACTUATORS

The robot is assembled with standard off-the-shelf R/C servomotors as actuators. This kind of servo has an integrated closed loop position control circuit which detects the *pulse-code modulated signal* that emanates from the controller board for commanding the servo to a

[3] http://www.aibo.com/

given position. In this implementation, each servo is commanded to a given position by the robot control program by addressing it an integer value within the interval {0, 255}. Two different sizes of servomotors are used for ELVINA. For the four ankle and hip joints we use the stronger ones with an output torque of 8.8 kgcm and for the other eight joints we use servomotors with an output torque of 3.9 kgcm.

2.3 POWER SUPPLY

Since the actuators are very energy consuming, the power supply of the robot turns into a delicate problem. The power source must meet requirements such as high capacity, low weight and reasonable costs. Although the controller board can provide power at a constant voltage, a separate power circuit for the actuators is preferable. Noise and power glitches produced by the high currents of these components must not be allowed to interfere with the controller board circuits. The robot is equipped with four 1.2 volt, 1700 mAh NiMH cells as power source for the actuators and a single 9 volt alkaline battery for the controller board, altogether with a weight of 150g. This gives the robot an effective operating time of approximately twenty minutes.

2.4 CONTROLLER BOARD

The robot has the EyeBot MK3[4] controller onboard, carrying it as a backpack. The EyeBot MK3 consists of a 32-bit micro-controller board with a graphics display and four push buttons for user input. The camera is directly connected to the controller board without a frame grabber. The EyeBot MK3 also has a serial communications interface. The robot control programs are developed on a host computer. After a cross-compilation they are downloaded, in executable code format, to the EyeBot controller. The serial line is then only used for uploading experimental data to the host computer since all signal processing is carried out on the EyeBot controller itself.

2.5 VISION SYSTEM

Vision is the most important sensor of this robot. Therefore, it is equipped with a full color 24 bit digital camera, which is based on CMOS technology. The camera is directly connected to the controller board, and physically attached to an actuator on top of the robots torso. This arrangement gives the camera, relatively to the robot's body, one degree of freedom in the horizontal plane and a camera sweep angle of 85 degrees.

2.6 INFRARED PSD RANGE SENSOR

A single camera cannot be used to accurately measure the distance to a nearby object. This is instead achieved with a near-infrared PSD range sensor, which consists of an IR emitter and a position sensitive detector in a single package. The principle of this sensor is based on triangulation, which means that the sensor is relatively insensitive to the texture and color of the object at which it is pointed. The emitter, placed below the detector in the package, illuminates a small spot on an obstacle with modulated IR light. A lens forms an image of the spot on the active element at the back of the detector. The output of the detector element is a function of the position on which the image is falling [Jones et al 1999]. Within the range of about eight to 40 centimeters distance to the object, a value of sufficient accuracy (resolution < 1 millimeter) is produced.

2.7 FIRMWARE AND SOFTWARE

The EyeBot MK3 controller board is running an operating system that consists of two main parts, the RoBIOS (robot basic I/O system) and the HDT (hardware description table). The same RoBIOS is shared by all hardware configurations of a robot controlled by an EyeBot, but the HDT differs to account for different sensors or actuators connected to the actual hardware. Each actuator has a unique workspace according to its position on the robot [Ziegler et al, 2001]. The workspaces for all actuators are specified in the HDT file by setting suitable values.

In order to control the movements of a limb, the partial movements of all involved joints must be coordinated and synchronized to get the desired motion. For this reason, a servo locomotion module has been developed. The idea behind it is that a complete gait cycle is specified by nine integer-valued vectors, each of the vectors specifying given positions of the robot's limbs. Each of the vectors consists in fact of a set of control parameters, a position value for each actuator and two time constants. The first of the vectors in the set correspond to the robot's initial position and the second vector corresponds to the second position of the robot and so on. By interpolation from the values of one vector to the values of another vector the robot's limbs is caused to smoothly move from one position to another position. The first time constant specifies how fast the limbs should move between consecutive positions and the other constant specifies time delay before the position of the limbs is updated. To obtain a complete gait cycle the set of vectors specifying it is interpolated once and the robot is made to continuously walk by iteration. All robot control programs that we have developed is implemented in C language.

3 EVOLUTIONARY ALGORITHM

The evolutionary algorithm used is a steady state evolutionary strategy [Banzhaf et al, 1998], running on the robot's onboard computer. A population that stems from a manually developed individual is created with a uniform distribution over a given search range. Then four individuals are randomly selected from this initial population. These individuals are evaluated and their

[4] http://www.ee.uwa.edu.au/~braunl/eyebot/

fitness is measured. The two individuals with the better fitness values are considered as parents and the two individuals with the lower fitness are replaced by the offspring's of the parent individuals. The selection, evaluation and reproduction phases of the evolutionary strategy is then repeated until the maximum number of trials is reached.

3.1 INITIALIZATION

The initial population is composed of 30 individuals with 126 genes randomly created with a uniform distribution over a given search range. The 126 integer-valued parameters used as genes are listed in table 1 together with their initial search range.

Table 1: Parameter List

Parameter type	Number of values	Unit	Initial Range	Search Range
Speed	9	Time (0.01s)	3-4	0-2
Delay	9	Time (0.01s)	20-200	0-150
Servo position	108	Angle (1/256)	4-253	0-2

The search range for each parameter type (e.g. speed, delay and servo position) is determined from experience in manually developing gaits. The search range, see table 1, is defined as the magnitude of the Euclidean distance between a certain gene in the manually developed individual and the corresponding gene in a randomly created individual. The search ranges are set to suitable values in order to produce a sufficient amount of individuals in the population that are capable of good performance in the evaluation. Especially when the servo position parameters are set to a wider range, many individuals resulted in the robot falling over almost immediately in the evaluation phase. When an initial population was created evolution began.

3.2 TOURNAMENT SELECTION

A tournament selection is used to select individuals for parents and the individuals to be replaced by their offspring. Four different individuals are randomly picked from the population and then evaluated one at a time. The two individuals who get the higher fitness are considered as parents and their offspring, produced by recombination and mutation, replaces the two individuals with the lower fitness in the population. The number of generations a certain individual can be selected to be in the tournament is unrestricted.

3.3 REPRODUCTION

For reproduction both mutation and recombination is used. Recombination takes the two individuals considered as parents, p_1 and p_2, and creates two child individuals, c_{1i} and c_{2i}. Each gene of the child c_{ki} then gets the value

$$c_{ki} = p_{ki} + \alpha_{ki}(p_{1i} - p_{2i})$$

where c_{ki} is the ith gene of the kth child individual, p_{ki} is the ith gene of the kth parent individual, p_{1i} and p_{2i} are the ith gene of the two parents p_1 and p_2. The α_{ki} is a number randomly chosen to be either –1 or +1.

In each of the child individuals produced, 20 % of the genes are mutated by a small amount. The genes in these two individuals are selected by random to undergo mutation and it is possible for a gene to be mutated several times. The gene to be mutated gets a value according to the equation

$$c_{ki,mutate} = c_{ki} + \delta_{ki} m_{ki}$$

where $c_{ki,mutate}$ is the mutated ith gene of the kth child individual, c_{ki} is the gene to be mutated. The δ_{ki} denotes a number randomly chosen to be either –1 or +1. The m_{ki} are a random number with uniform distribution that determines how much each gene should be mutated and it is set proportional to each parameter type's search range. That is, for the delay parameter, m_{ki} values are set to maximum 6% of its search range and for the speed and servo position parameters, m_{ki} values are, in a similar way, set to 33% maximum respectively.

4 EXPERIMENTAL METHOD

The aim in short term for our experiments is to optimize a set of integer values, used as control parameters for a biped robot gait. They should move the robot faster, straighter and in a more robust manner than the previously manually developed set of parameter values did. Our intention with this section of the paper is to describe how these experiments were performed and how individuals were evaluated.

4.1 EXPERIMENTAL SETUP

The experimental environment is shown in figure 2. The robot is placed on top of a table with a surface of relatively low friction during the evolution. A target wall of 50cm height and white color is placed at one end of the table and to mark the center of that end there is a vertical black stripe on the wall. Right above the robot (65cm above the table surface) there is a horizontal beam, used as a carrier for the power supply cables and for the security chain. In order to minimize the cable's influence on the robot during locomotion there is a counterweight connected to the cables via a string that is extended around a pulley. The purpose of this arrangement is that the counterweight drags the cables and the security chain as the robot moves forward.

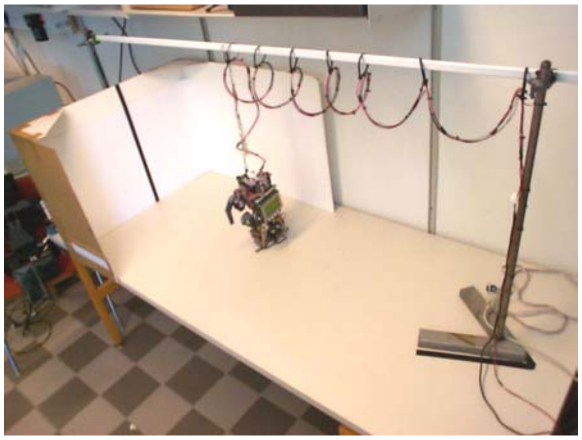

Figure 2. The experimental setup.

4.2 EVALUATION

Each individual evaluates under as equal conditions as possible. The robot‘s starting position is at a distance of about 40cm from the wall and facing it. The experimenter centers the robot according to the black line by using its onboard camera. Once centered, the robot measures its distance with the PSD infrared range sensor and starts to locomote towards the wall. After a fixed number of gait cycles it stops. Again it measures its distance from the wall and pans its head (camera) to search for the black line on the wall. Using these measurements and the time required for the locomotion trial, it calculates a fitness value for this actual individual. The robot is then manually reseted to its starting position by the experimenter for the next individual to be evaluated.

4.3 USE OF CAMERA

The primary task for the onboard camera is to provide a precise tool for determination of direction. Initially, the camera is set to continuos image mode, whereas the frames are put to the EyeBot‘s LCD screen and thus made visible to the experimenter. The robot is considered as centered when the image of the black line appears near the center of the LCD display. A single snapshot is then analyzed by the software image processing routines to precisely determine the robots position relative the black stripe. In detail, this is made by first converting the image to a grayscale image. Next, the Sobel operator is applied to each pixel of this image in order to perform an edge detect operation [McKerrow, 1991]. The originally color image is then converted into an ordinary bitmap image, where the edges of the black stripe produces two parallel, vertical lines. The average point, in horizontal direction, between those lines is then easily calculated by the software. This value is then stored for later use.

After an individual has performed a trial, the camera is again used to determine how straight the robot moved during the trial. As the robot body remains fixed the camera pans from left to right in steps, where it takes one snapshot between each step and analyze each of the images in the same way as described above. One step corresponds approximately to an angle of 3 degrees. For each snapshot the new calculated average point between the lines is compared with the earlier obtained average point and when the difference between these is less than 1, the camera stops moving. When the initial average point was calculated, the camera servo was in its middle position. The difference between the value obtained when the camera stops panning and the middle position value is considered as the angular deviation θ from the desired (straight) path of locomotion.

4.4 USE OF THE PSD RANGE SENSOR

While the robot uses its onboard camera for determination of direction, distances are measured using a near-infrared PSD range sensor located at the robot’s chest. Initially the robot is manually moved to a specific starting point, about 40cm from the target wall. At this fixed position, the start distance is determined by averaging six consecutive PSD sensor readings, with 0.2s interval between the readings. After an individual has performed a trial successfully (i. e. without falling) it stops and again uses the PSD sensor to determine its stopping distance from the wall.

The PSD sensor returns a raw value (between 0 and 999) that from about 10 to 40 cm is almost linear with distance. These raw values are easily converted into millimeters by creating an internal distance conversion table for the HDT. To do this, the robot was placed at a fixed distance from the target wall and slowly moved towards the wall while 128 sensor readings were taken and its distance from the wall was measured in millimeters. However, since the PSD sensor output is linear with distance and the resolution of the sensor is less than 1 millimeter in the actual range, the raw value was instead used in the fitness calculations.

4.5 FITNESS

To determine an individuals fitness score both its average velocity during the trial and its ability to move in a straightforward path is taken into account for. The fitness score function is defined as

$$score = v(d_0, d_f, t) \times s(\theta, d_f)$$

where $v\ (d_0,\ d_f,\ t)$ is the average velocity of the robot during the trial and $s(\theta,\ d_f)$ is the straightness function. The d_0 and the d_f denote the initial and the final distances to the target wall respectively and t is the time passed during the trial. The straightness function is dependent of both the angular deviation θ and the robot's final distance to the target wall and it is thus defined as

$$s(\theta, d_f) = \frac{d_f(f(\theta) - 1) + 150 - 10f(\theta)}{140}$$

$$f(\theta) = 1 - \frac{|\theta|}{128}$$

Here, $f(\theta)$ is a normalization function to convert θ into a 0 – 1 measure. The values 150 and 10 are used as constants for the straightness function because they are raw values corresponding to the maximum and minimum measurable distances for the PSD sensor. The straightness function accounts for the robot's final distance from the black target strip - with the robot at a fixed orientation θ being larger when the robot stops closer to the target wall. Finally, the average velocity function is defined as

$$v(d_0, d_f, t) = \frac{d_0 - d_f}{t}$$

In the case when an individual does not maintain the robot's balance during a complete trial (e.g. the robot falls) it receives a score of zero. This has to be done manually by the experimenter, since the experimental system is not equipped with any device for detecting that kind of occurrence. If the robot's gait causes it to turn so sharply that it cannot pan its head far enough to face the black target strip, the actual individual then automatically receives a zero fitness score.

5 RESULTS

For this evolutionary strategies experiment we used an initial population of 30 individuals and ran for nine generations. The best-evolved individual received a fitness score of 0.1707. The manually developed individual was also tested and received a fitness score, averaged over three trials, of 0.1051. The best-evolved individual outperformed the manually developed individual both in its ability to maintain the robot in a straight course and in robustness, i.e. with a less tendency to fall over. The qualities of different individuals were also tested in other ways than direct fitness measuring. To evaluate one generation, consisting of four individuals, took approximately 30 minutes in this experiment.

Table 1: Evolved Parameter List

Parameter type	Number of values	Unit	Initial Range	Evolved Range
Speed	9	Time (0.01s)	3-4	2-5
Delay	9	Time (0.01s)	20-200	42-300
Servo position	108	Angle (1/256)	4-253	3-253

5.1 FITNESS

In figure 3 the average fitness scores for each generation is shown as dots and the line is produced by statistical analysis, i.e. linear regression, of the dots. Since the slope of the line is positive, we observe a tendency towards better and better fitness values.

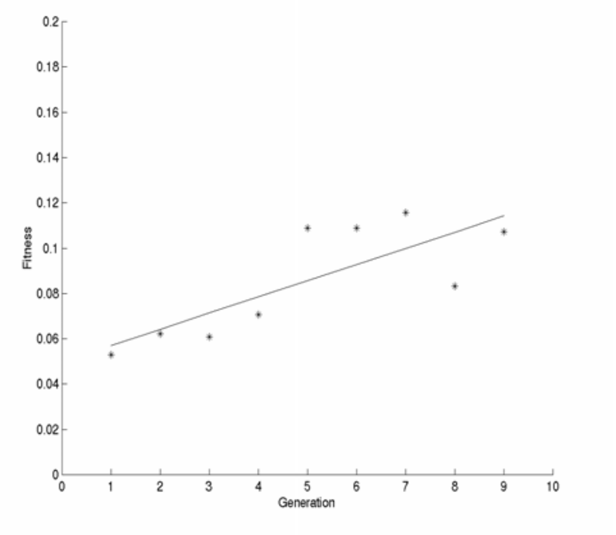

Figure 3.The Resulting Average Fitness Scores

5.2 SPEED

There was no improvement observed in the robot's speed of locomotion. The best-evolved individual and the manually developed individual could both move the robot in a speed of approximately 10.5cm/minute. They were tested over a 100cm run, which took 9.5 minutes each.

5.3 STRAIGHTNESS AND ROBUSTNESS

How well the individuals performed were tested both on the same flat table surface as the evolutionary experiment was made on, as well as on a high friction rubber surface.

On the low friction table surface the manually developed individual behaved unstable and was not able to keep the robot in a straight course, since it caused the robot to suddenly make sharp turns in an unpredictable way. This lack of accuracy decreased when the surface was changed to a high friction, rubber surface. The robot then proved to have a sidewise deviation of about 33cm on a 100cm distance. When the best-evolved individual was tested, it moved the robot forward on both types of surfaces with a sidewise deviation of less than 5cm on a 100cm distance.

6 DISCUSSION

An observation made when manually developing gaits was that, in order to determine if a certain set of gait parameters i.e. an individual is of sufficient quality, it must be iterated in more than one step. That is because the mechanical structure of the robot is non-rigid. When moving a limb (e.g. a leg), the trajectory, thus the limb's final position, is affected by from which position the movement started. How much the torso leans also affects the resulting position of the robot. Therefore, the final position of the robot's movements depends on if it was moved to the starting position by itself or if it was manually put there. To account for the behavior in the phase when the robot is moving from the last position in the first iteration to the initial position in the next iteration, each individual had to be iterated three times.

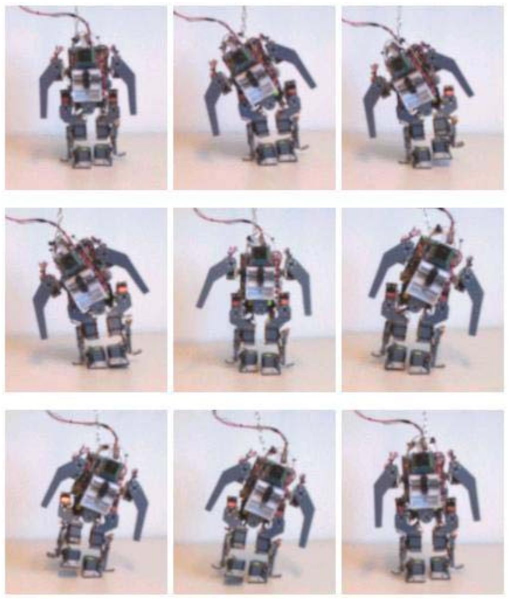

Figure 4. Series of pictures showing a complete gait cycle, from top left to bottom right.

Evolved parameters are showing good performance in some ways, yet not in others. In the beginning of the evolution, many individuals were moving the robot both fast and in a robust manner, except for in the single phase of moving the center of mass in order to stand on one leg and lift the other to take a step. The torso was not leaning enough towards the standing leg side, which resulted in that the robot fell when it lifted the other leg. An individual showing this tendency then received a zero fitness score. As the evolutionary process went on this type of behavior decreased and gradually died out.

To run an evolutionary experiment took about 30 minutes for each generation, so the 9-generations experiment that we present in this paper went on for almost five hours. Between each generation of four individuals evaluated, we paused the experiment for about 15 minutes in order to spare the hardware and especially the actuators. The main reason for this is that the actuators accumulate heat when they are running continuously under heavy stress. They then run the risk of getting overheated and gradually destroyed. We also observed that the position control circuit of the servomotors is sensitive to temperature. When commanding a servomotor to a given position by addressing it a fixed integer value within the interval {0, 255}, the physical angle of the servo's output shaft dislocates over time as the temperature of the servo increases. Since the robot's feet are coupled to each other via nine actuators their relative positions are then affected so much by this drift that it could cause the robot to fall. One way to handle this problem is, as mentioned above, to run the robot intermittent so that the servos maintain an approximately constant temperature.

Evolving efficient gaits with real physical hardware is a challenging task. In the six months of manually developing gaits and testing the evolutionary algorithm, frequent maintenance of the robot was indeed necessary. The torso and both the ankle actuators were exchanged once as well as the two hip servos. The most vulnerable parts of the robot were proved to be the knee servos. Both these servos were replaced tree times.

7 CONCLUSIONS

The work presented in this paper constitutes of two main parts, the construction of a small humanoid walking robot and a genetic programming experiment performed on it. By manually developing locomotion module parameters, the robot was made capable of autonomous static walking in a first stage. In the next stage we performed a genetic programming experiment on the robot in order to improve the manually developed gait. For this, we used a steady state evolutionary strategy that was run on the robot's onboard computer. This algorithm evolved an individual that outperformed the previously manually developed set of parameter values in a sense that it moved the robot in a straighter path and in a more robust way.

We believe that the ELVINA robot platform is capable of development. In future research we aim to improve the hardware and software of the ELVINA, resulting in a more robust and durable robot platform. Then it will be possible to do more genetic programming experiments in order to have the robot to accomplish tasks that are more complicated. Such could be using vision to navigate, collaborate and interact with other robots of this kind and balancing and walking on an inclined plane.

References

P. Nordin and M. G. Nordahl (1999). An evolutionary architecture for a humanoid robot. *Proceedings of the Fourth International Symposium on Artificial Life and Robotics (AROB 4th 99).* Oita, Japan.

P. Dittrich, A. Burgel and W. Banzhaf (1998). Learning to move a robot with random morphology. *Phil Husbands and Jean Arcady Meyer, editors, First European Workshop on Evolutionary Robotics* (pp. 165—178). Berlin: Springer-Verlag.

S. Nolfi and D. Floreano (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines.* Massachusetts: The MIT Press.

W. Banzhaf, P. Nordin, R. E. Keller and F. D. Francone (1998). *Genetic Programming~ An Introduction: On the Automatic Evolution of Computer Programs and Its Applications.* San Francisco: Morgan Kaufmann Publishers, Inc. Heidelberg: dpunkt verlag.

G. B. Parker and J. W. Mills (1999). Adaptive hexapod gait control using anytime learning with fitness biasing. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '99)* (pp. 519-524). San Francisco: Morgan Kaufmann Publishers, Inc.

N. Jakobi (1998). Running across the reality gap: octopod locomotion evolved in a minimal simulation. *Phil Husbands and Jean Arcady Meyer, editors, First European Workshop on Evolutionary Robotics* (pp. 39—58). Berlin: Springer-Verlag.

T. Gomi and K. Ide (1998). Emergence of gait of a legged robot by collaboration through evolution. *P. K. Simpson, editor, IEEE World Congress on Computational Intelligence.* New York: IEEE Press.

G. S. Hornby*, S. Takamura, O. Hanagata, M. Fujita and J. Pollack (2000). Evolution of controllers from a high-level simulator to a high dof robot. *J. Miller, editor, Evolvable Systems: from biology to hardware; proceedings of the third international conference (ICES 2000)* (Lecture Notes in Computer Science; Vol. 1801 pp. 80-89). Berlin: Springer-Verlag.

M. A. Lewis, A. H. Fagg and A. Solidum (1992). Genetic programming approach to the construction of a neural network for control of a walking robot. *Proceedings of the IEEE International Conference on Robotics and Automation.* New York: IEEE Press.

G. S. Hornby, M. Fujita, S. Takamura, T. Yamamoto and O.Hanagata (1999). Autonomous evolution of gaits with the Sony quadruped robot. *Proceedings of the Genetic and Evolutionary Computation Conference.* San Francisco: Morgan Kaufmann Publishers, Inc.

G. S. Hornby, S. Takamura, J. Yokono, O.Hanagata, T. Yamamoto and M. Fujita (2000). Evolving robust gaits with AIBO. *IEEE International Conference on Robotics and Automation.* pp. 3040-3045.

J. L. Jones, A. M. Flynn and B. A. Sieger (1999). *Mobile Robots: Inspiration to Implementation.* Massachusetts: AK Peters.

J. Ziegler, K. Wolff, P. Nordin and W. Banzhaf (2001). Constructing a small humanoid walking robot as a platform for the genetic evolution of walking. In print.

P. J. McKerrow (1991). *Introduction to Robotics.* Wollongong: Addison-Wesley.