Reinforcement Learning for a Biped Robot to Climb Sloping Surfaces

A. W. Salatian
Applications Engineer
National Instruments
Austin, TX 78730

Keon Young Yi
Dept. of Electrical Engineering
Kwangwoon University
Seoul 139-701, Korea

Yuan F. Zheng*
Dept. of Electrical Engineering
The Ohio State University
Columbus, OH 43210
e-mail: zheng@ee.eng.ohio-state.edu

Accepted October 29, 1996

A neural network mechanism is proposed to modify the gait of a biped robot that walks on sloping surfaces using sensory inputs. The robot climbs a sloping surface from a level surface with no priori knowledge of the inclination of the surface. By training the neural network while the robot is walking, the robot adjusts its gait and finally forms a gait that is as stable as when it walks on the level surface. The neural network is trained by a reinforcement learning mechanism while proportional and integral (PI) control is used for position control of the robot joints. Experiments of static and pseudo dynamic learning are performed to show the validity of the proposed reinforcement learning mechanism.

*To whom all correspondence should be addressed.

Journal of Robotic Systems 14(4), 283–296 (1997)

CCC 0741-2223/97/040283-14

センサー入力を使って斜面を歩行する二足歩行ロボットの歩き方を改善するために、ニューラル・ネットワーク機構を提案する。ロボットは、地面の傾斜に関する事前情報なしで、水平面から斜面を登坂する。ロボットの歩行中にニューラル・ネットワークを訓練することで、ロボットは歩き方を調整し、最終的に水平面で最も安定した歩行を行うようになる。PI（比例＋積分）制御を使ってロボット・ジョイントの位置制御を行うと同時に、ニューラル・ネットワークは強化学習機構によって訓練される。静的または疑似的な力学学習の実験を行い、今回提案した強化学習機構の有効性を確認する。

1. INTRODUCTION

Legged robots have better mobility on rough terrain than wheeled robots since they can use isolated footholds that optimize support and traction. Moreover, the payload can be moved smoothly, despite pronounced variations of the terrain, by using an active suspension that decouples the path of the body from the paths of the feet. Because of these reasons, legged robots are very useful in replacing human beings in extreme environments such as in nuclear power plants and on ocean floors. Of all the legged robots, biped robots most closely resemble the build of human beings. This makes biped robots suitable for environments that were originally created for human beings to access. A great deal of research has been conducted to make biped robots useful.[1–8] However, it is difficult to maintain stable locomotion while the robot is walking on different surfaces.

Recently, many researchers have studied biped robots walking on different terrains.[9–11] When a biped robot walks on a surface other than level floors, the robot must use different gaits to maintain stability. Previous works used sensors to detect terrain conditions, and modified the gait accordingly. These approaches must deal with many problems such as sensing the terrain, planning a path, selecting a foothold, and adjusting step length, etc.

In our previous work,[9] we enabled a biped robot to climb a sloping surface with unknown degree of inclination by using on-and-off sensors installed underneath the feet. The sensors were used to make sure that the feet were completely in contact with the ground. By examing the orientation and position of the feet with respect to the rest of the biped robot, the degrees of inclination of the sloping surface can be calculated. For the calculated slope, a stable gait can be designed, based on the robot kinematics and dynamics, for walking on the slope. Because the approach involves substantial computation in real-time, the robot must walk slowly when the degree of inclination varies. Alternatively, expensive computing equipment must be involved.

In this article, we present a neural mechanism to control the biped robot. The neural mechanism generates a new gait gradually and automatically based on the forces measured by the force sensors installed underneath the feet. Because the force sensors serve as an indicator of the stability of the biped robot, it is not necessary to calculate the degree of inclination and re-program the gait accordingly. The joint positions of the robot are adjusted until the force sensors indicate that the robot has a stable gait. The relationship between the force and the adjustment of the joint positions is highly nonlinear. A neural network is naturally a good choice to map the relationship between the two.

In reviewing the existing literature, we found very few papers that report the use of neural networks for controlling biped locomotion. Kun and Miller[12] used a CMAC neural network for the adaptive control of side-to-side and front-to-front balance, as well as for maintaining good foot contact of a biped robot. The CMAC neural network generates required motion of the robot including the position of the hips, and the amplitude and velocity of the side-to-side lean, etc., from the desired step parameters such as step length and step rate. By using the CMAC network, complex computation of robot kinematics and dynamics in real-time becomes unnecessary. For training the CMAC neural network a supervised learning process is used.

In our work, we use unsupervised reinforcement learning as opposed to supervised learning. The reason for using unsupervised learning is that the neural network receives no direct instruction on which joint position needs to be modified. Moreover, the network is not sure in which direction the selected joint should change its position. Every joint of the robot is associated with a neuron called a *joint neuron*. Every joint neuron is further attached with two pairs of neurons called *direction neurons*. All the neurons possess certain values called *neuron values*. During the learning process, a joint neuron with the maximum neuron value is selected to modify the position of its corresponding joint, and likewise a direction neuron is

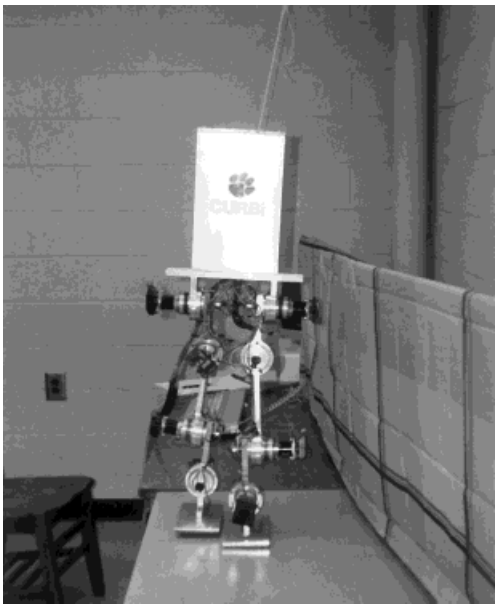

Figure 1. The SD-2 robot.

selected to determine the direction of modification. If the selected joint and direction neurons result in a correct modification of the motion, i.e., the robot becomes more stable, we reinforce the selection by increasing the neuron values. Otherwise, the neuron values are reduced. By using this rewarding and penalizing mechanism, the neural network converges quickly to a correct set of neuron values that will generate a stable gait for the sloping surface. The unsupervised reinforcement learning method is attractive because the method does not require explicit feedback signal. The learning process proceeds online, and the computation involved in the process is simple.[13] Furthermore, noise characteristics of the sensors are taken into account in the learning process.[14]

In the following section we introduce the structure of the biped robot and its gait. In the third section a reinforcement learning mechanism for the biped robot to walk on the sloping surface will be developed. The fourth section will be devoted to the discussion of the experiments including static learning and pseudo dynamic learning. The article is summarized in the fifth section.

2. THE SD-2 BIPED ROBOT

The target of this study is a biped robot called SD-2 (Fig. 1).[4] In this section we will describe the structure of the SD-2 robot and the static gait that the robot uses to walk.

2.1. The Structure of the SD-2 Biped Robot

The SD-2 robot has nine links and eight joints as depicted in Figure 2. Four joints control the motion in the sagittal (fore-and-aft) plane, and the other four control the motion in the frontal (left-and-right) plane. Each leg has four degrees of freedom (DOF). The top two joints of each leg emulate the hip joint, while the bottom two are for the ankle joints. Note that the robot has no knee joints. The structure of the biped robot is symmetrical. That is, there is no structural difference between the front and back sides of the robot. The front side of the robot is arbitrarily selected, and once selected, it is marked with a ''CURBI'' sign (see Fig. 2). Likewise, the foot is symmetrical and there is no structural difference between the toe and heel of the foot, so called because they are on the front or back side of the robot, respectively.

Figure 3 shows the foot which has three contact points to the ground. To implement the neural control mechanism, a pair of force sensors, TK-13-T092P-10C transducer class strain gauges from the Micro-Measurements Division of Measurements Group Inc.,[15] are mounted on two of the contact points, toe and heel, respectively.

2.2. The Gait of the SD-2 Biped Robot

A gait that is statically stable for the SD-2 robot is shown in Figure 4. In the figure, the dotted squares represent the swining foot, and the big dots represent the projection of the center of gravity (COG). There are eight static configurations within a complete step, which are called primitive points (PP). The joint motions of the robot are linear between two consecutive PPs. For convenience, each piece of the linear motion is called a phase. Each phase is again decomposed into a large number of setpoints. A setpoint is a set of joint positions. The controller moves the joints to a new setpoint every 28 ms. When the biped robot completes a step, it goes through eight phases or 2,000 setpoints (see Fig. 5).

At the beginning of walking (home position), the projection of the COG is at the center of the two-foot supporting area. In phase 1, the joints in the frontal plane are moved to shift the projection to the left foot. The joints in the sagittal plane are rotated to transfer the projection forward. In phase 2, the right leg swings forward using the hip joint in the sagittal plane, and the ankle joint of the left leg and the hip joint in the sagittal plane are moved to transfer the COG forward. At the same time the hip joints in the frontal plane are rotated to lift the leg; this is required

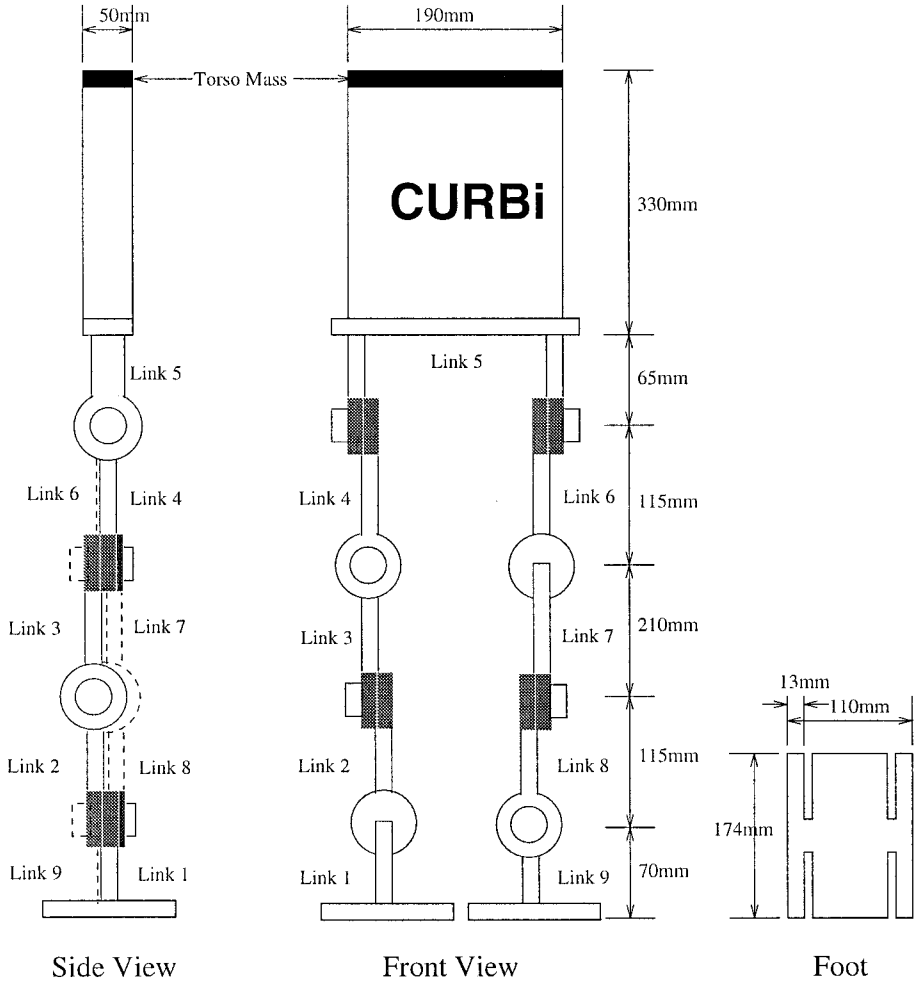

Figure 2. The structure of the SD-2 robot.

because the robot has no knee, and lifting the leg eliminates the possible collision between the foot and the floor. At the end of this phase the swinging foot is parallel with the floor. In phase 3, joints in both the sagittal and frontal planes are moved simultaneously to make the right foot touch the floor while the COG remains over the left foot. In phase 4, all the joints are rotated to shift the projection of the COG to the center of the two-foot supporting area again. These four phases are the first half of the step. The same procedure are repeated for the next half of the step, phase 5 through phase 8, which has the right foot support the robot.

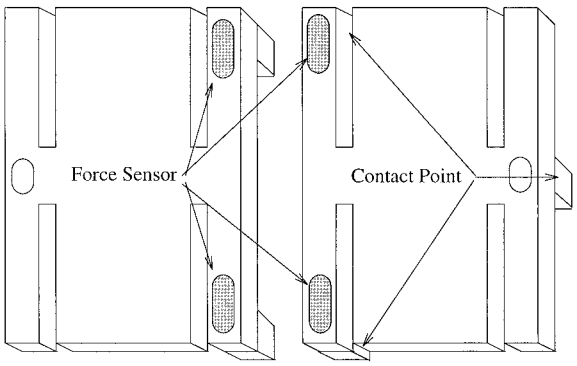

Figure 3. The foot and force sensor.

The position of the projection of the COG (dots in Fig. 4) can be obtained using the following equations:

$$A_{cog} = L\frac{F_{lt} * 1.5 + F_{lh} * 0.5 + F_{rt}}{F_{tot}}, \quad (1)$$

$$B_{cog} = L\frac{F_{lt} * 1.5 + F_{lh} * 0.5 + F_{rt} * 2 + F_{rh}}{F_{tot}}, \quad (2)$$

$$F_{tot} = F_{lt} + F_{lh} + F_{rt} + F_{rh}. \quad (3)$$

where A_{cog} is for phases 8 through 3, and B_{cog} for phases 4 through 7. F is the force, and the subscripts of F, l,

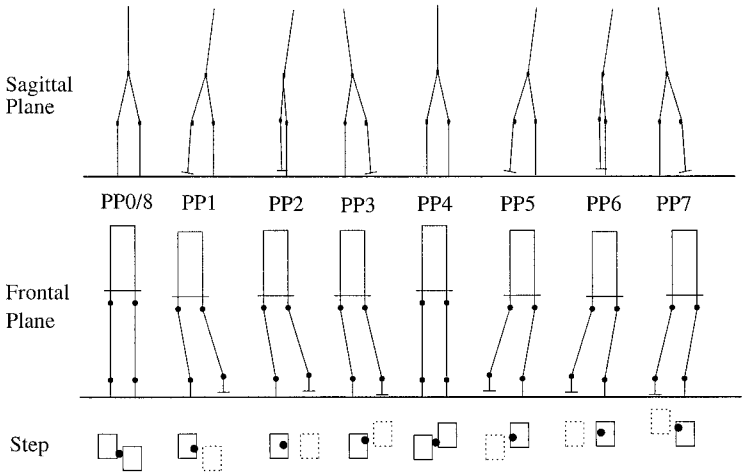

Figure 4. The gait on a level surface.

r, h and t represent ''left'', ''right'', ''heel'', and ''toe'', respectively. L is the length of the foot. The origin of the coordinates is defined at the heel of the right foot. The above equations are suitable when the robot uses a static gait, and the balanced forces are evenly distributed underneath the surface of the foot.

The actual position of the projection of the COG can be obtained by adding up the counts of steps, which should be increased at the end of phase 8. Figure 5 shows the projection measured when the biped walking on a level floor and a sloping floor (+5°: solid line), respectively. Note that X/L in the figure means the length in the frontal plane normalized by the foot length L of the robot. The offset between the two lines indicates that the robot is not so stable when it walks on the sloping surface because the projection of the COG is close to the edge of the supporting area, or the stability margin is reduced. It is necessary to adjust the gait so that the robot can be as stable as it walks on a level floor. This process is called the *stabilizing process* in the rest of this article, which is the goal of our research.

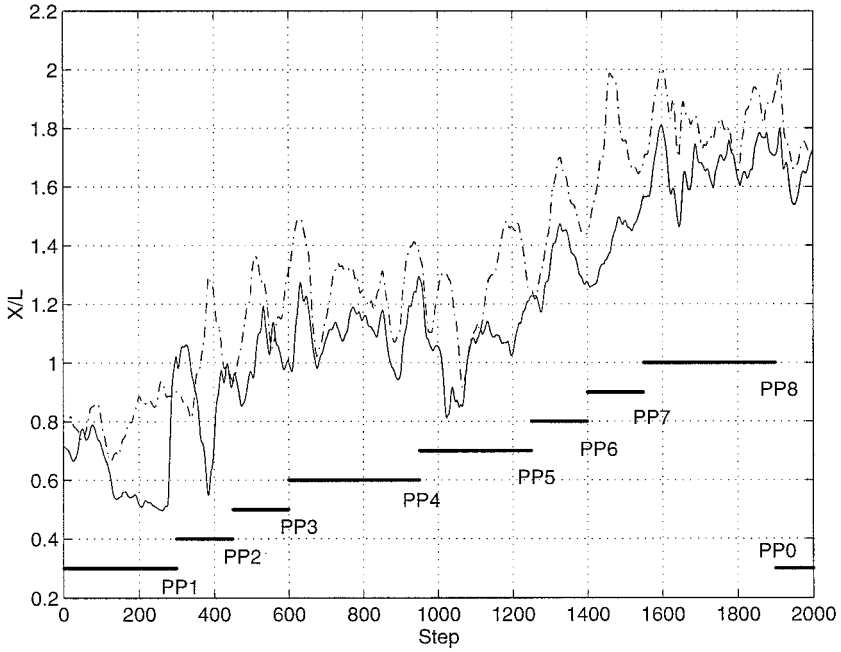

Figure 5. The projection of the COG for a level floor (–.–) and a 5° slope (——).

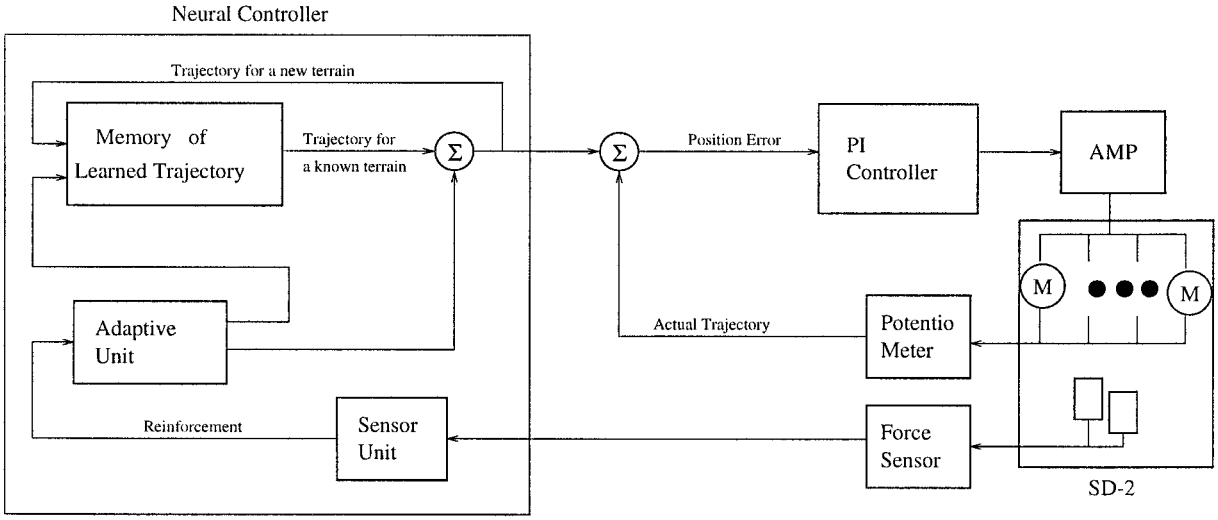

Figure 6. Configuration of the control system.

3. NEURAL CONTROLLER

3.1. Control System Configuration

To control the SD-2 biped robot, we designed a control system. The configuration of the control system is shown in Figure 6. In the figure, the PI controller uses conventional proportional and integral control to drive the eight joint motors. The neutral controller consists of a memory, which stores programmed or previously learned gaits, and an adaptive unit (AU). The AU is responsible for modifying joint trajectories such that the robot can walk on a new surface. Both the neural controller and the AU are implemented in a personal computer (PC). To interface the signals between the PC and the amplifier/sensor, a DDA-08 Metrabyte digital-to-analog converter and a DAS-8 Metrabyte analog-to-digital converter are installed inside the PC. The amplifier is built with a single component power operational amplifier, Apex Microtechnology PA02, for each joint. This amplifier can drive up to 12 volts and 2 amperes in each direction of motion for every joint. The potentiometer is used to measure the joint angle, and the power sources for sensors are isolated from the amplifier to avoid interference.

3.2. Neural Mechanism

The main part of the neural controller is the AU, which is responsible for modifying the joint motions. The AU consists of multiple neurons with inhibitory or excitatory synaptic contact between them (Fig. 7). The circles in Figure 7 represent the neurons, the white squares at the end of the line connecting neurons represent excitatory contacts, and the black squares represent inhibitory contacts.

The numbered circles are called the joint neurons. Each joint neuron corresponds to one joint to be modified for each PP. Because the motions of the joints in the frontal plane of the robot are the same for walking on a level surface or on a sloping surface, the neural network is only responsible for modifying the joint motions in the sagittal plane. Recall that there are four joints in the sagittal plane and eight PPs in each step. It appears that $4 \times 8 = 32$ joint neurons are

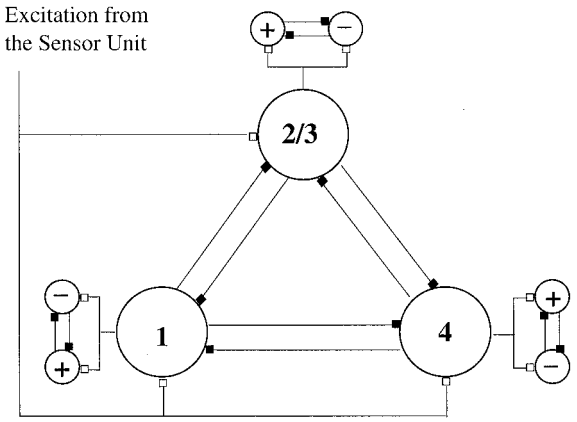

Figure 7. Excitation/inhibition model of the neural network.

necessary, but we use only one neuron to modify the motions of the two hip joints. That is, the modifications to the two hip joints will always be the same. This is because the two hip joints are aligned on a single axis (see Fig. 2). Only when they rotate simultaneously and by the same amount can the torso of the robot rotate without alternating the position of the swinging leg. As a result, the motions of the torso and the legs are decoupled. This decoupling is important for the generation of a new gait, which represents a minimal modification to the original gait. Consequently, we use only $3 \times 8 = 24$ joint neurons.

The neuron that is active during a certain period of time will allow the position of the corresponding joint to be modified. Every time, only one neuron is active. This is why the contact between the joint neurons is inhibitory. The reason for modifying only one joint is due to the unsupervised learning. Because the network does not have a teacher, we can test only one joint at a time to see if the selected joint is effective in stabilizing the robot. If it is, the selection is reinforced; otherwise, the selection is penalized. If more than one joint is modified, the judgement for reinforcement will be impossible to make.

There are two pairs of smaller circles attached to each joint neuron (only one pair is shown in Fig. 7) which are the direction neurons (24 pairs of neurons in total). The first pair is active when the biped climbs up a slope, and the second pair becomes active when the biped climbs down. Only the pair designated for climbing up is shown in the figure. One of the two direction neurons is denoted with a ''+'' sign and the other is with a ''−'' sign. They determine the direction in which the corresponding joint should modify its motion. When the ''+'' neuron is active, the modification is to increase the joint angle, and vice versa. The direction neurons are activated by the joint neuron but deactivate each other. This is the reason why the contacts between the two direction neurons are inhibitory.

The network works in the following way: Suppose that the biped is climbing up a slope. At the end of each phase the Sensor Unit (SU) reads the signal from the force sensors, and calculates the projection of the COG using (1), (2), and (3). If the result indicates that the robot is not so stable in comparison with the desired projection of the COG. The SU sends excitatory signals to the three joint neurons in the AU. One of the three neurons is activated, and the other neurons enter the inactive mode; this selection is based on the neuron value of each neuron. Once excited, the neuron, assumed 3 here, sends an excitatory signal to the two direction neurons attached to it. Suppose the neuron ''+'' is activated first; this selection is similar to that of the joint neuron. The ''−'' neuron enters the inactive mode. In consequence, the angle of the joint 3 will be increased. This process continues until no excitatory signal is sent from the SU, indicating that the biped has reached desired stability. Now, the AU becomes inactive and the joint trajectories for the next phase are obtained from the memory of the learned trajectory.

For every joint j, the associated joint neuron has a neuron value of $w(j, pp)$ at primitive point pp(j = 1, 2, 3 and $pp = 0, 1, 2, \ldots, 7$). $w(j, pp)$ is updated by a neuron learning algorithm that will be discussed in detail in the next subsection. The goal of this learning algorithm is to reward the neuron by increasing $w(j, pp)$ if the modification to its corresponding joint improves the stability of the robot. Otherwise, $w(j, pp)$ is reduced. As a result, the joint neuron that is effective in stabilizing the robot will have better chance to be selected in the next step of training. As the learning process continues and converges, the network will always choose the right joint to modify such that the stability of the robot is improved quickly when the robot climbs a slope from a level surface.

The same principle is used to train the direction neurons. The direction neurons at a primitive point (pp) have a neuron value of $v(ud, jn, h, pp)$ where the undefined parameters are $ud = 0, 1$ indicating whether the robot is climbing up (1) or down (0) the slope, jn being the joint number, and $h = 0, 1$ denoting the direction of the joint angle modification with 0 for negative and 1 for positive directions.

The neural learning mechanism as discussed above is called reinforcement learning, so called because the selection of an effective neuron in the learning process is reinforced. The principle of reinforcement learning has been well studied by Barto and other researchers.[13,14,16] The details of the principle will not be addressed any further here. In the next subsection, the details of the algorithm that we use to train the neural network are presented.

3.3. The Reinforcement Learning Algorithm

We use the difference between the forces exerted at the toe and exerted at the heel as the reinforcement signal, i.e.:

$$\Delta f = f_{heel} - f_{toe}, \tag{4}$$

where f_{heel} represents the forces sensed from heels and f_{toe} from toes. It is easy to see that this signal has the same characteristics as the projection of the COG, but

the computation is simpler. For a new sloping surface, the robot is as stable as desired if $\Delta f = \Delta f_{bal}$ where Δf_{bal} is the idea force difference. Δf_{bal} can be obtained by recording it when the robot walks on a level surface provided that the gait is optimal. Designing an optimal gait for walking on a level surface is relatively simple. The training algorithm is listed below.

STEP 0: Determine the current primitive point number pp, where $pp = 0, 1, \ldots, 7$.

STEP 1: Set all the neuron values to zero if this particular pp had no previous learning (1st round of training).

STEP 2: If $|\Delta f - \Delta f_{bal}| < \varepsilon$, no further training is required. Otherwise, the learning process as described in the following steps becomes necessary. Note that ε, a small positive constant, can be determined in accordance with the level of the sensor noise.

STEP 3: Set ud using the following equations:

$$ud = 0 \quad \text{if} \quad \Delta f > 0 \tag{5}$$

$$ud = 1 \quad \text{if} \quad \Delta f < 0 \tag{6}$$

STEP 4: Define $yw(j)$ as

$$yw(j) = w(j, pp) + noise, \; j = 1, 2, 3. \tag{7}$$

where the noise is an artificially introduced random variable distributed uniformly on an interval $[-1, 1]$. The value of $yw(j, pp)$ will be used to choose the joint whose position is to be modified. From (7), it can be seen that a neuron whose $w(j, pp)$ is high tends to have a high $yw(j, pp)$. For two neurons whose $w(j, pp)$ values are close, the neuron with a smaller $w(j, pp)$ may still be picked by the addition of a random noise as in (7). This gives the smaller neuron a fair opportunity of being selected because it might be more effective in stabilizing the robot.

STEP 5: Set the joint number jn to j that is associated with the largest $yw(j)$. jn defines the joint angle that is chosen by the network to be modified.

STEP 6: Define $yv(h)$ as

$$yv(h) = v(ud, jn, h, pp) + noise, \; h = 0, 1 \tag{8}$$

STEP 7: Set the direction dir to h that is associated with the largest $yv(.)$.

STEP 8: Define $\Delta q(jn)$ according to the two decisions made above. For example if $jn = 3$, and $dir = 1$, then $\Delta q(3) = +dq°$, where dq is the small positive angle.

STEP 9: Compute the new force difference and the reinforcement signal z.

$$\Delta f°_{old} = \Delta f°_{new}, \tag{9}$$

$$\Delta f°_{new} = \Delta f_{bal} - \Delta f, \tag{10}$$

$$z = \Delta f°_{old} - \Delta f°_{new}. \tag{11}$$

STEP 10: The neurons are trained using following equation:

$$v(ud, jn, dir, pp) = v(ud, jn, dir, pp) + c_1 * sign(s_1 * \Delta f°_{new}) * z, \tag{12}$$

$$w(jn, pp) = w(jn, pp) + c_2 * sign(s_2 * \Delta f°_{new}) * z \tag{13}$$

where c_1 and c_2 are positive constants servings as the learning rates, and s_1 and s_2 are the saturation indices. The index will be set to 1 if the neuron value of the corresponding neuron is less than the predefined threshold, S_1 or S_2; otherwise it is set to 0.

STEP 11: Goto Step 2.

The above training continues until it is automatically stopped, that is, $|\Delta f - \Delta f_{bal}| < \varepsilon$ becomes true. The whole training process is then called one round of training. In Step 10, the value of each neuron will be increased if the second term of the right-hand side is positive. This means that the neuron is rewarded because the modification is right, i.e., $\Delta f°_{new} = \Delta f_{bal} - \Delta f$ is positive and decreasing, and the reinforcement signal $z > 0$. The neuron value can be reduced as a punishment if the above two conditions are not true.

Saturation indices limit the upper bound of the neuron value, i.e., no further reward is given to the neuron when it has a relatively large neuron value in Step 5 and Step 7. This limit gives no undesirable effect to the learning mechanism. In other words, if one neuron has reached the threshold value, no other neuron will be selected. However if more than one neuron has reached the threshold value, one of those neurons can still be selected by using (7) and (8). That is, the random value of the noise will determine the selection.

Once all the above steps are completed, Δf is close to the desired one, and all the neurons will have acquired some neuron values. When the trained network is used for stabilizing the robot again, it should take less time to reach the goal than the previously untrained network. After training the neural

network for a necessary number of rounds for each PP, the AU permanently stores the new set of primitive points along with the degrees of inclination of the slope in the memory (if the degree of inclination can be measured). The next time the robot runs into a slope with the same degree of inclination (if it is given to the robot), the controller will take this learned set of PPs and generate an appropriate gait. Consequently, the robot can perform walking without the involvement of the AU.

4. EXPERIMENTS

We performed two experiments, static learning and pseudo dynamic learning, to show the validity of the proposed mechanism. In static learning, the neural network is trained for a single primitive point until the robot reaches the desired stability for that primitive point. Before the training is completed, the robot does not move to another primitive point. In pseudo dynamic learning, the neurons are trained while the robot is walking. The neuron values are trained for every setpoint and the modification is made to the current PP. Because the training is conducted while the robot is in motion, the term ''dynamic'' is used. To avoid being confused with dynamic walking of the robot, the learning is called pseudo dynamic learning.

In the two experiments, we employed only one joint neuron for two hip joints, for the reason presented in the previous section. In pseudo dynamic learning, there is a possible joint motion discontinuity when the neural network is switched to new neurons for the next PP because the new neurons were not trained during the current phase. To prevent this motion discontinuity, we enable the learning mechanism only during the first half of the phase, and the motion adjustment accumulated in the training will be made to the current PP. In the second half of the current phase, the displacement made by the neural mechanism is linearly reduced to zero such that the robot reaches the next PP smoothly.

4.1. Static Learning

The biped walks on a 3° slope using the gait designed for level surface. For simplicity, we concentrate on the second PP of the trajectory. According to Figure 5, the vertical projection of the COG at the second PP is at 0.85. Apparently, the robot is not very stable. We should adjust the gait using the mechanism proposed until the projection of the COG becomes above $X/L = 1.1$. The reason for $X/L = 1.1$ is because when the robot is in phase 2, this gives the best stability. The training coefficients used were $c_1 = 0.025$, and $c_2 = 0.008$, which were decided experimentally. Small training coefficients require longer training time, and large values cause undesirable oscillation. The coefficients for STEPs 2 and 8 were $dq = 0.015$ and $\varepsilon = 10$ (note that the unit for ε is 0.1 newton). Saturation indices were limited by $S_1 = 10$ and $S_2 = 10$, respectively.

In Figure 8 the projection of the COG and the reinforcement signal for the first round of training are depicted. One can see the changes for both. Figure 9 shows the joint position changes and the neuron values of the joint neurons for the first round of training. Note that the training takes 1850 steps. As each step lasts 28 ms, the training takes about 5.18 s. Since two hip joints, 2 and 3, are linked and controlled by one neuron, the changes for joint 3 are omitted. Note that the joint position changes stay near zero at the very beginning (the first 1000 steps). This indicates that the network is not sure what modification is the best for the current sloping surface. One may also notice that every joint is adjusted until 1500 steps of training are done, after which joint 1 is selected as the most efficient joint to regain a stable gait (joint 1 neuron is rewarded big).

Figure 10 shows the joint position changes and the neuron values of the joint neurons for the first round of training when the biped was on the 7° slope. Figure 11 shows the case that was experimented on the 5° slope. Once again the robot is at the second PP of the trajectory but has been trained on 3° and 7° slopes. One may see that only joint 1 (top graph on the left side of Fig. 11) has joint position changes. This means that only joint 1 is selected for regaining the stability. In Figure 11, the initial neuron values of the joint neurons are started with those acquired during the training for the 7° slope. Note that in the figure the neuron value of the joint 1 neuron is limited by the saturation index, which is 10. Other neuron values are not changed because they are smaller than that of the joint 1 neuron. As a consequence, only joint 1 is adjusted, and the stable gait is regained very quickly (the total step count is reduced from 1850 to 550, i.e., the adaption time is reduced from 5.18 s to just 1.54 s).

To investigate the characteristics of the neural network, we repeated the training five times. Every time the robot was put on the slope with a gait only suitable for a level floor, but the neuron values were set based on the previous training.

Figure 12 shows the result of the fifth round of training. While it might not be very clear in the first

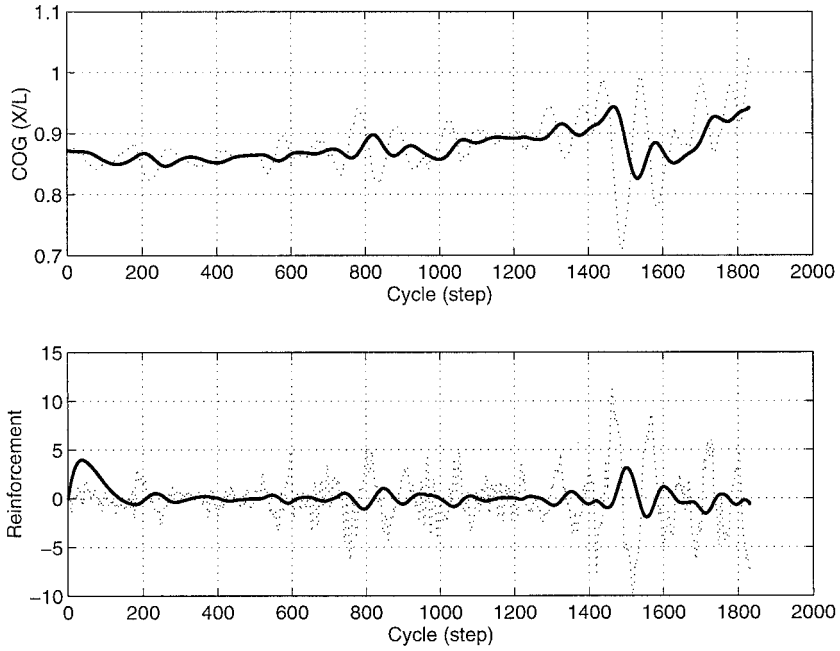

Figure 8. The vertical projection of COG and the reinforcement signal for the first round of training (3°).

few rounds of training how the projection of the COG is going to change, it clearly goes toward 1.1 at the end of the fifth round of training. As mentioned in the previous experiment, the AU needs less time to stabilize the robot when we use the trained neurons (the fifth round needs only 0.42 s). One strange thing is that we had two modified stable gaits for this PP. One gait adjusted the ankle, and the other bent the torso. The result was unpredictable, but once the joint, hip or ankle, was selected at the beginning, the gait

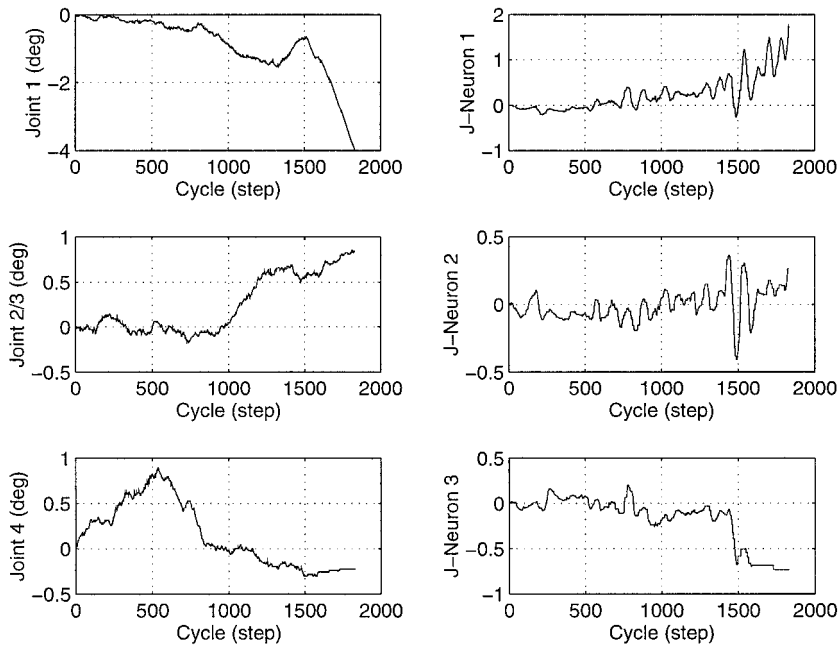

Figure 9. The adjustment of the joint positions and the neuron values of the joint neurons in the first round of training (3°).

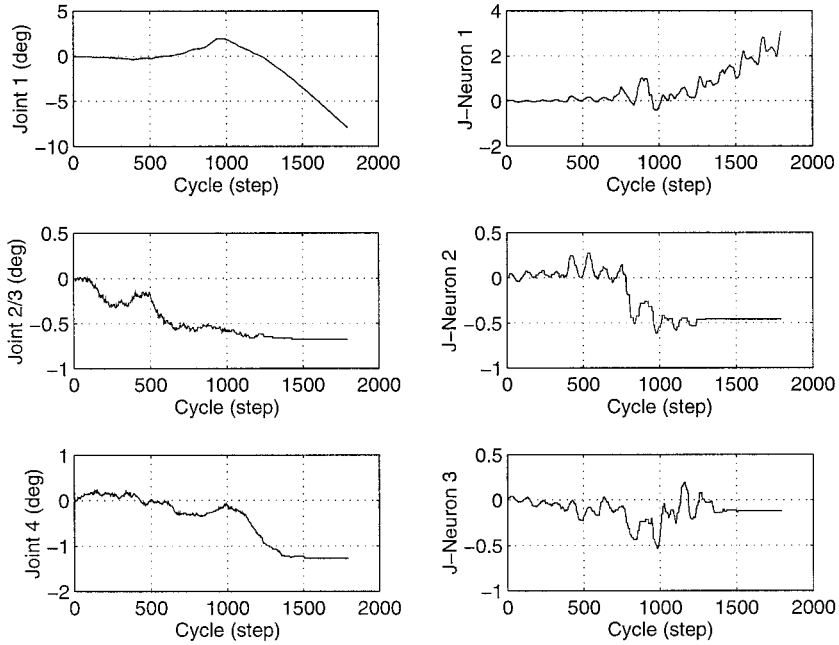

Figure 10. The adjustment of the joint positions and the neuron values of the joint neurons in the first round of training (7°).

was determined. This means that joint selection depends on the initial condition of the biped, but once a joint is selected and rewarded as a consequence of adjusting, this joint has a greater chance of being selected. This phenomena is natural because human beings also use both ways to stabilize a gait. Figure 13 shows the stable postures of the robot after one round (top of Fig. 13) and five rounds (bottom of Fig. 13) of training, respectively, for both cases. Note that after five rounds of training the biped has less modi-

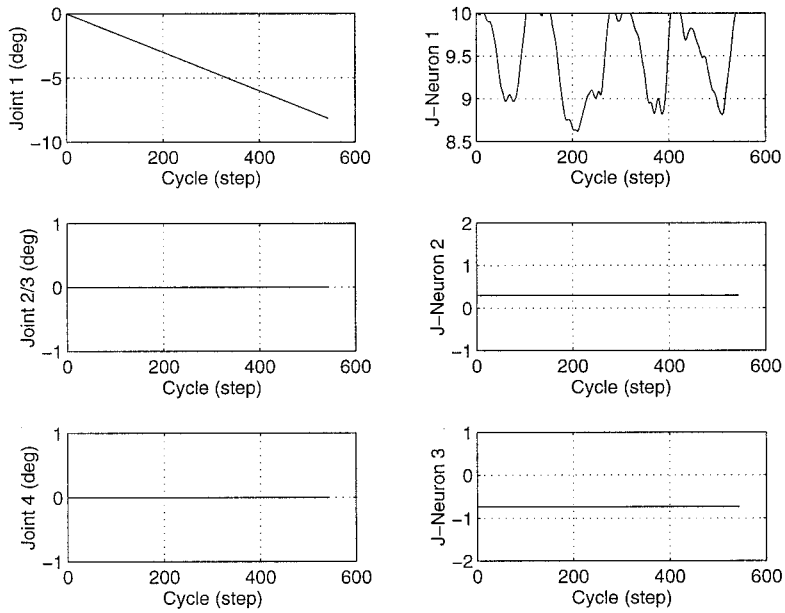

Figure 11. The adjustment of the joint positions and the neuron values of the joint neurons in the first round of training with *a priori* training (5°).

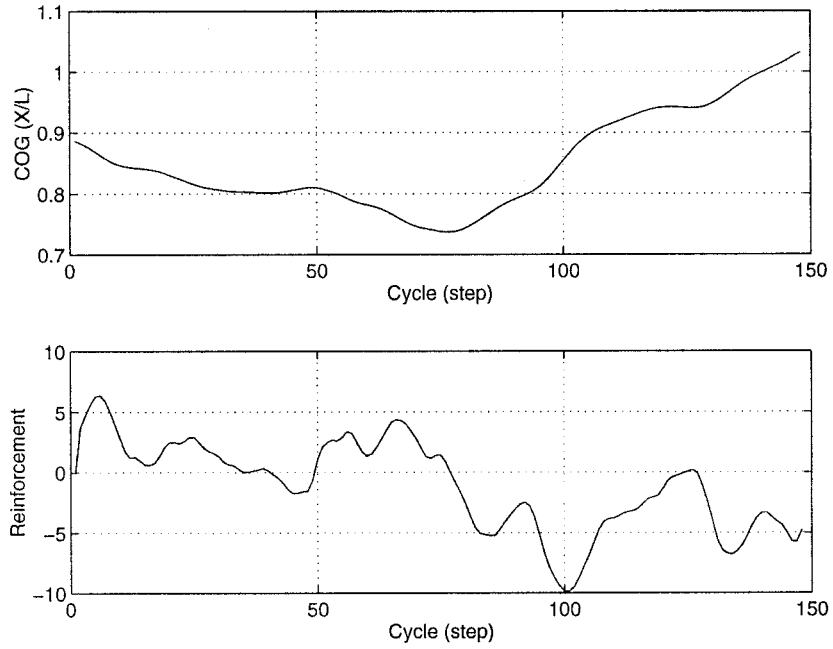

Figure 12. The projection of the COG and the reinforcement signal for the fifth round of training.

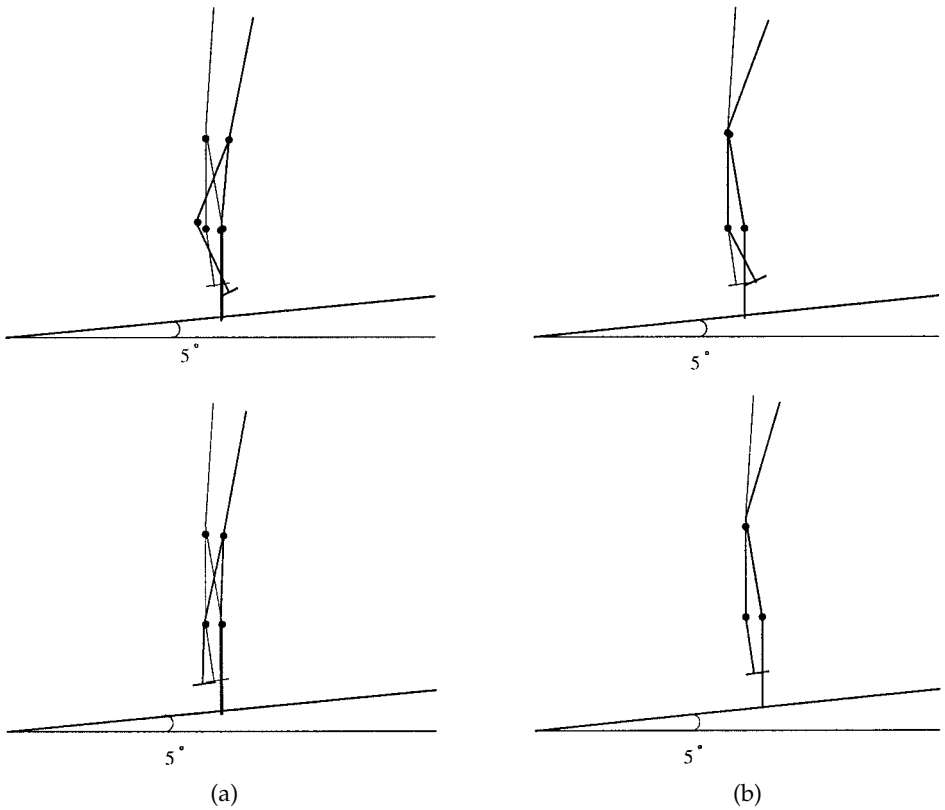

Figure 13. The balanced postures of the robot. (a) The adjusted ankle. (b) The bent torso.

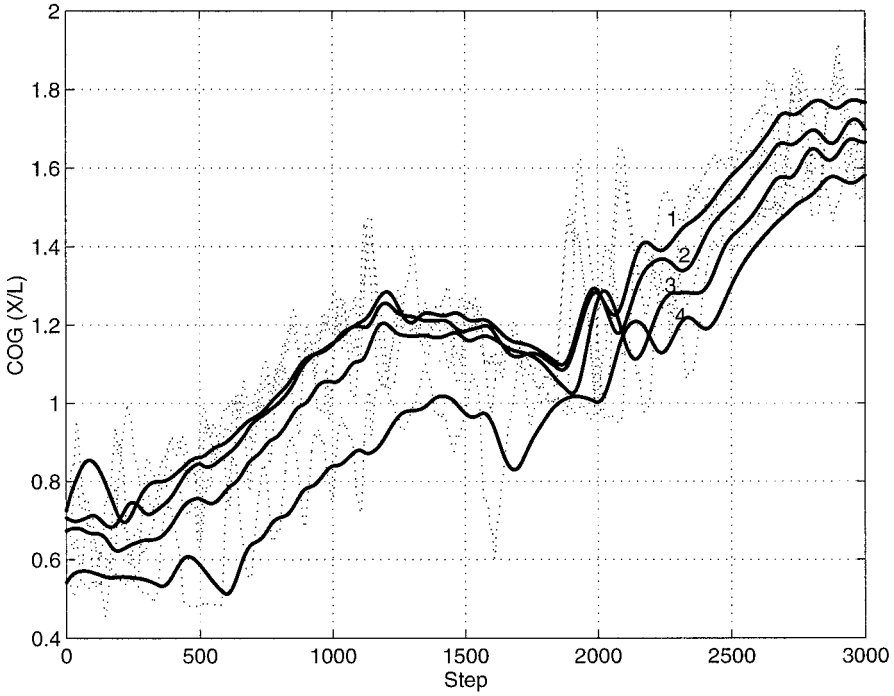

Figure 14. The projection of the COG for different rounds (the solid lines are results of a low-pass filtering to the actual data).

fication than after one round of training. This indicates that the modification of the joint becomes more efficient as the neural network has experienced more training. We also put the trained robot on a 20° slope. Again, the robot stabilized its gait quickly and effectively.

4.2. Pseudo Dynamic Learning

The experimental condition was the same as in the static learning experiment except that the neurons were trained while the biped was walking. In the experiment, the joint neuron corresponding to the ankle of the swinging leg was disabled to avoid any motion modification to the ankle. The foot may collide with the floor if the ankle joint is allowed to be adjusted.

We performed two experiments with previously trained and non-trained neurons, respectively. In the first experiment, a biped walked on the 5° slope with no previous training. The coefficients used were $c_1 = 0.025$, $c_2 = 0.008$, $dq = 0.02$, $S_1 = 10$, $S_2 = 10$, and $\varepsilon = 10$.

In the first round the biped was walking on a level floor (line 1 in Fig. 14). Then we let the biped walk on a 5° slope without adjusting the gait (line 4 in Fig. 14). Starting from the third round we enabled the AU, i.e., the training process was engaged. Note that as the training goes on, the lines of the COG become closer to the desired line (line 1 in Fig. 14).

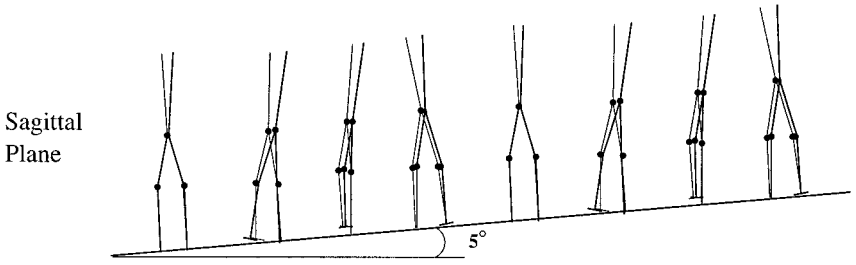

Figure 15. The gait on a 5° floor after 20 rounds of learning (thin line: gait for a level floor).

Lines 3 and 2 represent the result after 10 and 20 rounds of training, respectively. The gait of the robot, after the 20 rounds of training, is shown in Figure 15. It shows how the new gait compares with the gait programmed for the level surface.

5. CONCLUSIONS

A neural controller enabling the biped robot to walk on sloping surface is proposed and implemented. Unsupervised reinforcement learning is used to generate new gaits for the biped robot walking on sloping surface with the unknown degree of inclination. Static learning and pseudo dynamic learning are demonstrated to prove that the proposed mechanism is valid for robot to negotiate unknown or new sloping surfaces. Experimental results prove that it is possible for a biped robot to walk adaptively on unknown terrain using the neural network approach. Because the real terrains are more complex than what have been tested in our experiments, more studies need to be conducted to make biped robot walk robustly on different terrains.

This work was supported by NSF under grant IRI-9024499 and by a Seed Grant of the Ohio State University.

REFERENCES

1. H. Hemami and B. F. Wyman, ''Modeling and control of constrained dynamic systems with application to biped locomotion in the frontal plane,'' *IEEE Trans. Autom. Control,* **AC-24**(4), 526–535, 1979.
2. H. Miura and I. Shimoyama, ''Dyanamic walk of a biped,'' *Int. J. Rob. Res.* **3**(2), 60–74, 1984.
3. R. Katoh and M. Mori, ''Control method of biped locomotion giving asymptotic stability of trajectory,'' *Automatica,* **20**(4), 405–414, 1984.
4. Y. F. Zheng and F. Sias, ''Design and motion control of practical biped robots,'' *Int. J. Rob. Autom.,* **3**(2), 70–78, 1988.
5. N. Wagner, M. C. Mulder, and M. S. Hsu, ''A knowledge based control strategy for a biped,'' *Proc. IEEE Int. Conf. Rob. Autom.,* Philadelphia, 1988, pp. 1520–1524.
6. S. Kajita, T. Yamaura, and A. Kobayashi, ''Dynamic walking control of a biped robot along a potential energy conserving orbit.'' *IEEE Trans. Rob. Autom.,* **8**(4), 431–438, 1992.
7. J. Furusho and A. Sano, ''Sensor-based control of a nine-link biped,'' *Int. J. Rob. Res.,* **9**(2), 83–98, 1990.
8. E. R. Dunn and R. D. Howe, ''Towards smooth bipedal walking,'' *Proc. IEEE Int. Conf. Rob. Autom.,* San Diego, CA, 1994, 2489–2494.
9. Y. F. Zheng and J. Shen, ''Gait synthesis for the SD-2 biped robot to climb sloping surface,'' *IEEE Trans. Rob. Autom.* **6**(1), 86–96, 1990.
10. S. Kajita and K. Tani, ''Study of dynamic biped locomotion on rugged terrain,'' *Proc. IEEE Int. Conf. Rob. Autom.,* Sacramento, CA, 1991, pp. 1405–1411.
11. J. K. Hodgins and M. H. Raibert, ''Adjusting step length for rough terrain locomotion,'' *IEEE Trans. Rob. Autom.,* **7**(3), 289–298, 1991.
12. A. Kun and W. T. Miller, ''Adaptive dynamic balance of a biped robot using neural networks,'' *Proc. IEEE Int. Conf. Rob. Autom.,* Minneapolis, MN, 1996, pp. 240–245.
13. A. G. Barto, ''Some learning tasks from a control perspective,'' University of Massachusetts, COINS Tech. Report 90-122, 1990.
14. V. Gullapalli, J. A. Franklin, and H. Benbrahim, ''Acquiring robot skills via reinforcement learning,'' *IEEE Control Syst. Mag.*, February, 13–24, 1994.
15. Measurements Group, Inc., *Engineering Data Sheet,* Raleigh, NC, 1991.
16. A. G. Barto, R. S. Sutton, and C. W. Anderson ''Neuronlike adaptive elements that can solve difficult learning control problems,'' *IEEE Trans. Syst. Man Cybern., SMC-13*(5), 1983.
17. M. J. Mataric, ''Interaction and intelligent behavior,'' Ph.D. dissertation, MIT, Cambridge, MA, 1994.
18. V. Gullapalli, R. A. Grupen, and A. G. Barto, ''Learning reactive admittance control,'' *Proc. IEEE Int. Conf. Rob. Autom.,* Nice, France, 1992, pp. 1475–1480.