

# **A Hexapod Robot and Novel Training Approach for Artificial Neural Networks**

Nick Vallidis

## 1.0 Introduction

Mobile robots are finding their way into a number of tasks that are unfit for humans. Such tasks include those which are extremely repetitive, require great accuracy, or need to be performed in hazardous environments. For these reasons many wheeled vehicles have been designed and developed. More recently there has been an explosion of legged robot designs. These have many advantages over their wheeled counterparts. They allow the traversal of rougher terrain and can continue to function even after losing use of a leg (if so designed). These features are very useful in hazardous environments where maintenance cannot be performed.

Many researchers have explored the area of legged robots. Both statically stable (e.g., Ambler [9]) and dynamically stable (e.g., Raibert's hopping machines [10]) walking robots have shown great abilities. As more work is done in the field, there is a growing interest in looking to biology for solutions to problems raised by these complex robotic systems. This has led to a number of robots that take both mechanical designs and control strategies from biological legged-creatures. There have been a number of hardware systems modeled after insects, such as the walking-stick insect (e.g., [1]). Insects have also been studied for control strategies, resulting in systems using neural networks [2] and layering of behaviors [7,8].

The low weight-to-power ratio of most mechanical actuators causes trouble when designing walking robots. The muscles of insects are lighter, and capable of doing more work, than any existing mechanical actuator [1].

Control of legged robots is difficult, and so a machine-learning solution is needed. One machine-learning method for legged robots, which has great potential, uses artificial neural networks. Neural networks have the advantage of being modeled on the basic mechanisms responsible for biological control mechanisms. They are also well suited for the walking task, as they allow for automated adjustments of the control parameters.

Further advantages are found in the natural parallelism that exists in neural networks—allowing distributed computing hardware. Distributing the computing hardware adds to the robustness of the system. However, as was shown in research performed at Case Western Reserve University, there is also a natural robustness in properly evolved neural networks even without using parallel hardware [3].

There are many difficulties to be overcome in neural networks before they will be truly useful to robotics. No algorithmic method is known to develop the proper topology for a network to control a legged robot. A small network with too few connections might not be able to represent the behavior that is desired. If the network is made to be large and fully connected, the training time and number of training examples needed increases greatly, which is also highly undesirable. This leads us to another problem with neural networks—generating training data. The more complex the network, the more training data needed and the more work that is necessary before one can use the neural network. Currently the best promise comes from copying the topology of neural networks from organisms [2].

In this paper we will discuss the design and implementation of one specific robotic system—Black Widow. Black Widow is a six-legged robot that has been designed using many ideas inspired by biological walkers.

The resulting hardware platform is trained to walk over rough terrain while keeping its body level. The ability of walking robots to do this is one of their great advantages. It makes walking robots suitable as platforms for a variety of sensors as well as for other purposes in

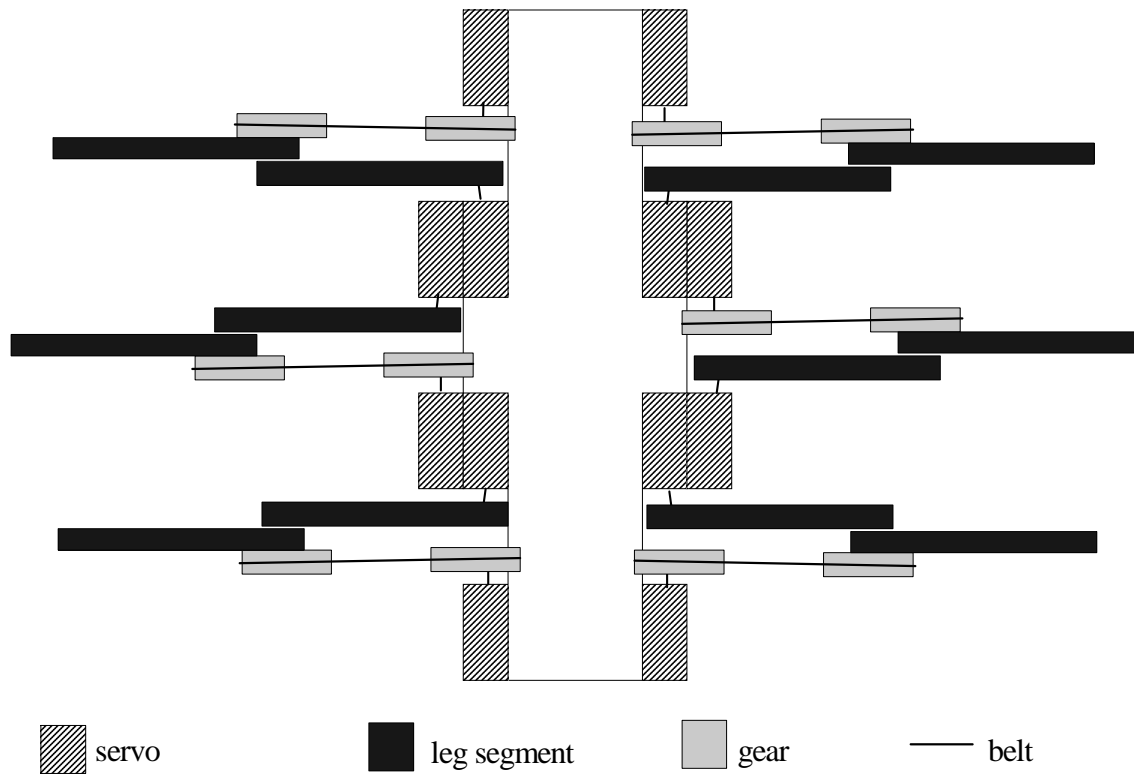
which a level surface is necessary (e.g., carrying tanks filled with liquids, or even as a vehicle to transport people).

In Section 2 we discuss the mechanical structure and physical capabilities of Black Widow. From there we move on to a discussion of the control strategies used on Black Widow in Section 3. Section 4 is a discussion of the results of our tests, and Section 5 presents the conclusions and discussion on directions for future work.

## 2.0 Hardware

### 2.1 Philosophy

When designing the hardware for Black Widow we tried to copy features of biological walkers—more specifically, insects. This allowed us to have a model to take ideas from, and provided a goal to strive for. Insects are much more capable of traversing varied terrain than any walking robot that has been built to date. As anyone who has ever watched an ant or a spider knows, there is almost no obstacle that can stop these creatures. Their walking is very robust, and they are able to quickly recover, even when they do make mistakes.



**Figure 2.1:** View from above of Black Widow

In many areas of the mechanical structure of Black Widow (see Figure 2.1) the biological influences are apparent. The first of these is the number of legs on the robot. There are six legs, which allows for a number of advantages. The robot can be statically stable with only three legs on the ground allowing for a very fast walking gait referred to as the "tripod gait". In this gait,

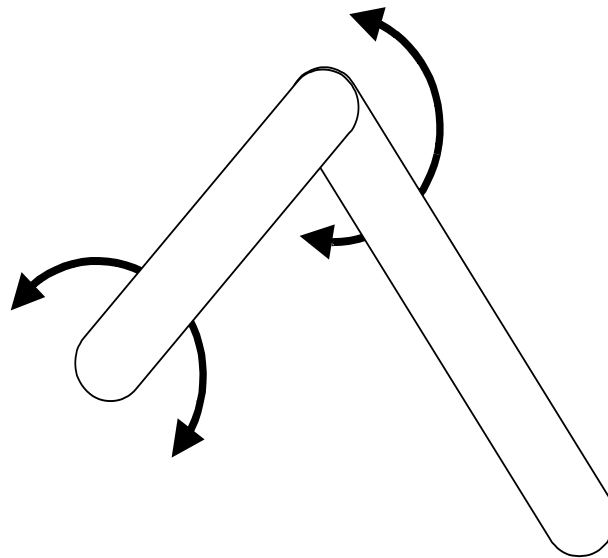
there are three legs on the ground propelling the robot forward (stance) while the other three legs recover to the front of their motion range (swing). The presence of six legs also allows for the robot to lose a leg (or the use of a leg) and still be able to walk using a different gait that keeps more legs on the ground.

Next, the relative lengths of the upper and lower segments of the legs of Black Widow are also modeled after insects. As in most insects, the length of the upper leg is 0.7 times the length of the lower leg. An effect of this length ratio is that the robot's body "hangs" from its knees. In other words, it is designed to walk with its knee joint above its hip joint having the effect that the body's weight is suspended from the legs.

The leg joints of Black Widow are based on those of the walking-stick insect. For example, the walking-stick insect has three degrees of freedom in its legs. It has two revolute degrees of freedom which allow motion in a plane (the knee and the hip) and a third revolute degree of freedom which allows the plane's angle to be changed with respect to the insect's body (perpendicular to the horizontal plane) [4]. Black Widow lacks this third degree of freedom, but we decided it was necessary to leave this out of the design in order to keep the weight of the machine down. Black Widow's two joints move independently of one another. The independence of the joints means that the joint angle of one leg segment is not limited by the position of the other, as is in the case of some legs used on walking machines. The pantograph structure is a common example of this limitation.

## 2.2 Mechanical Structure

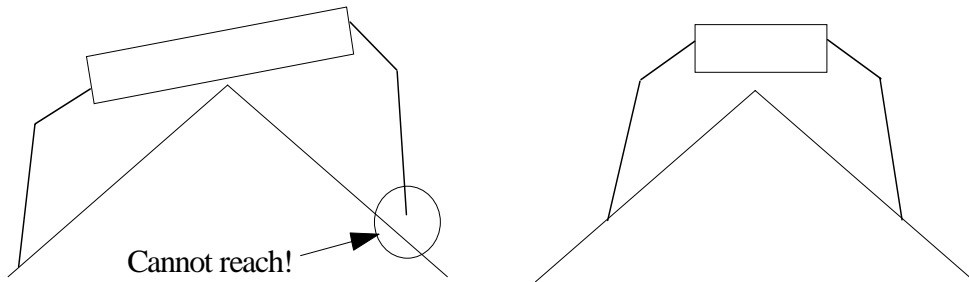
The six legs are arranged on the body so as to minimize the overall outer dimensions of the body. This positioning means that the center leg on both the front and the rear of the body are located further from the center line of the body than those on either side.



**Figure 2.2:** The two degrees of freedom of Black Widow's legs

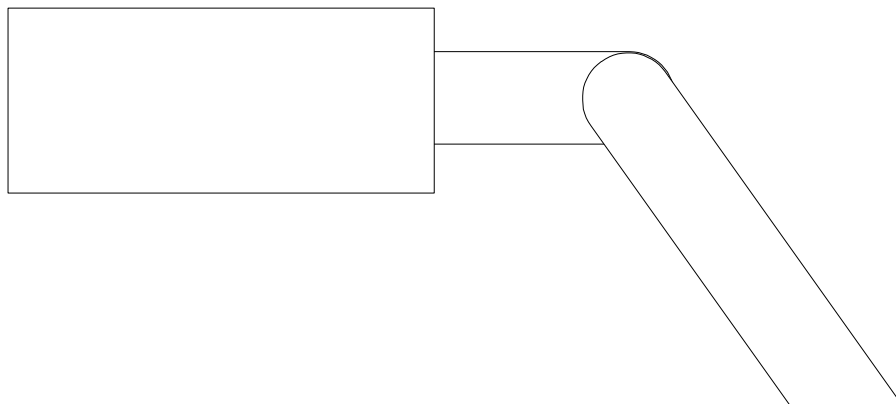
Black Widow's two degrees of freedom per leg (see Figure 2.2) are controlled by hobby servomotors. The upper leg segment is attached directly to the output shaft of the servomotor. The lower leg section is connected through a pair of gears and a belt to the servomotor. One gear

is attached to the output shaft of a servomotor and the other is attached to the lower leg. The gears are the same size so there is a 1:1 ratio between the servomotor and the lower leg. This design allows the motions of the upper and lower legs to be independent. As such, it is much easier to control, as the movement of one segment of the leg does not affect the angle of the other. In the case of upper leg segment motion, the lower leg segment's position is affected although its angle to the body remains constant.



**Figure 2.3:** The robot on the left with the elongated body cannot traverse as steep a slope as the one with the shorter body on the right (given that they have the same leg length)

As a result of the design of the legs (with only 2 degrees of freedom instead of 3), Black Widow does not walk as might be expected. It walks sideways, if compared to the motion of an ant. The front of the body is either side with 3 legs on it, and the sides of the body are those with no legs. Since the robot is symmetric, it is completely arbitrary which side is the front and which is the back. The resulting body is not very long, but quite wide. The wide body shape allows easier slope transitions (see Figure 2.3). The legs do not have to be as long, and for a robot of a given size, it is able to traverse larger slope transitions. Obviously this ability is paid for in other areas of performance, since Black widow cannot traverse a similarly shaped side slope. In view of the direction in which Black Widow walks, it was necessary to orient the body in this way.

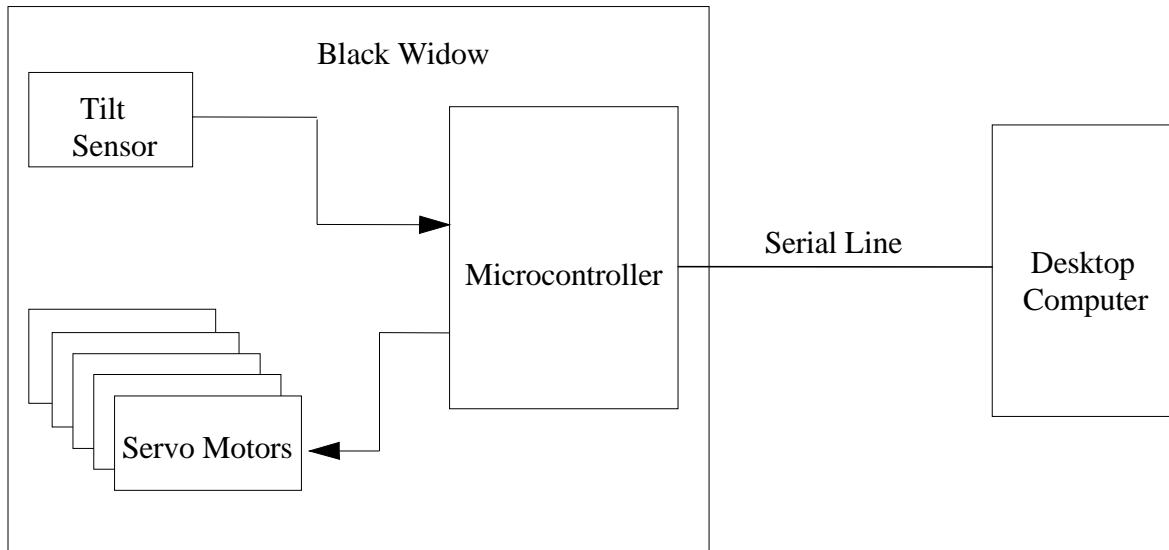


**Figure 2.4:** This is the position of the upper leg segment that requires the servo motor to do the most work. Therefore the maximum torque of the motor limits the length of the upper leg

The size of each leg was determined by two factors. The first was the ability of the leg to lift half of the body's weight, when extended straight out from the body (see Figure 2.4). The second was the length-ratio characteristic that was previously mentioned.

## 2.3 Control Hardware

Black Widow's control hardware is comprised of two computer systems communicating over a serial link (see Figure 2.5). The on-board computer is a microcontroller board that sends



**Figure 2.5:** The Control Hardware of Black Widow

commands to the 12 servomotors and reads the tilt sensor. The other computer is a desktop machine that handles all the high-level control, sending leg-position commands to the on-board computer and requests for the body tilt from the on-board computer.

In the future, the system will also incorporate another tilt sensor to measure the roll of the body (the current one only measures the pitch), and foot sensors will measure the force on each leg to distribute the weight of the body among the supporting legs.

## 3.0 Control and Learning

In this section we discuss the general use of neural networks and then discuss our own use. We also cover generating training data for the network and how the network was constructed and tested.

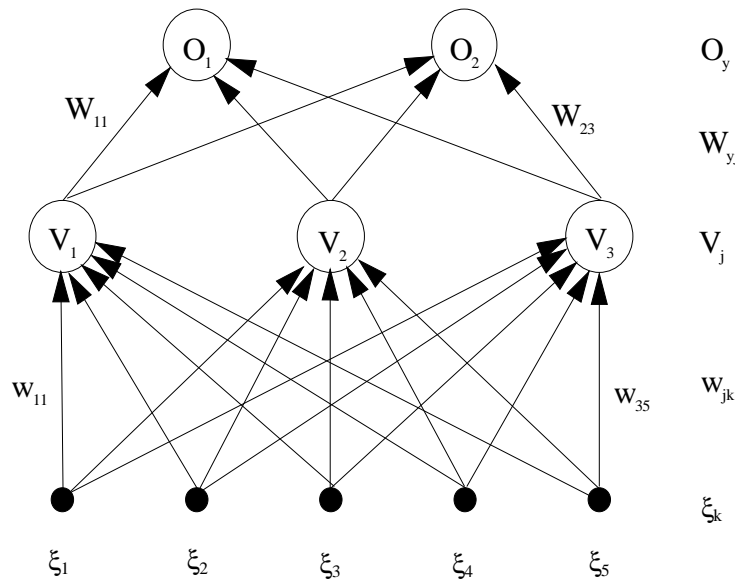
### 3.1 Artificial Neural Networks

The process of using ideas from biology extends to the control of Black Widow. Artificial neural networks provide the means of controlling the robot. Artificial neural networks have many advantages, as well as problems to face. They have the advantage that it is not necessary to find a model for the system one is controlling. Artificial neural networks find their

own model, through training, which could have nothing to do with the way the system actually works. Another advantage is the possibility of distributing the computation over several hardware systems, thus further copying the control systems of biological walkers. There are, however, problems with artificial neural networks. They need a great deal of training data to learn from. The goal is to have a machine, whose controller is based on artificial neural networks, which can learn in real time from its experiences. Until then, it is desirable to have a maximum amount of automation in the learning process.

Beer developed a recurrent neural network based on studies of the American Cockroach [5]. This network was hand-tuned to produce the desired results. Although a successful method, it also includes too much human interaction, as no automated learning occurs.

As a result of our requirements, we chose to use a multi-layer, feed-forward network, using the back-propagation algorithm for learning. This also includes a momentum term, to aid in getting out of local minima. The general form for this type of network can be seen in Figure 3.1. In this network there is one input layer (the  $\xi$ s), one hidden layer (the Vs) and one output layer (the Os).



**Figure 3.1:** a typical feed-forward network showing notation conventions  
Source: [Introduction to the Theory of Neural Computation](#) [12]

Ordinary back-propagation uses a formula of the form

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} = \eta \sum_{\mu} [\zeta_i^{\mu} - O_i^{\mu}] g'(h_i^{\mu}) V_j^{\mu}$$

where  $W_{ij}$  represents the weight we are updating,  $E$  the error function

$$E[w] = \frac{1}{2} \sum_{\mu i} [\zeta_i^{\mu} - O_i^{\mu}]^2$$

$h$  the learning rate,  $\zeta$  the desired output,  $g'(h)$  the derivative of the activation function  $g(h)$ ,  
 $g(h) = \tanh(\beta h)$

and  $V$  the output of the previous layer [12]. Momentum is calculated by adding a portion of the  $\Delta W_{ij}$  from the previous iteration of the back-propagation algorithm. The addition of the momentum term to the back-propagation algorithm results in the following modified equation

$$\Delta W_{pq}(t+1) = -\eta \frac{\partial E}{\partial W_{pq}} + \alpha \Delta W_{pq}(t)$$

where  $\alpha$  is the momentum parameter (ranging from 0 to 1) and  $t$  represents time [12]. This forces movement on the error surface to continue in approximately the same direction and with the same speed as was moved previously. The result is faster traversal along steep gradients in the error surface, leading to a quicker descent to the nearest minimum and also allowing the possibility of jumping out of a local minimum, if the gradient is steep enough approaching that minimum.

### 3.2. Training Data Generation and Network Topology

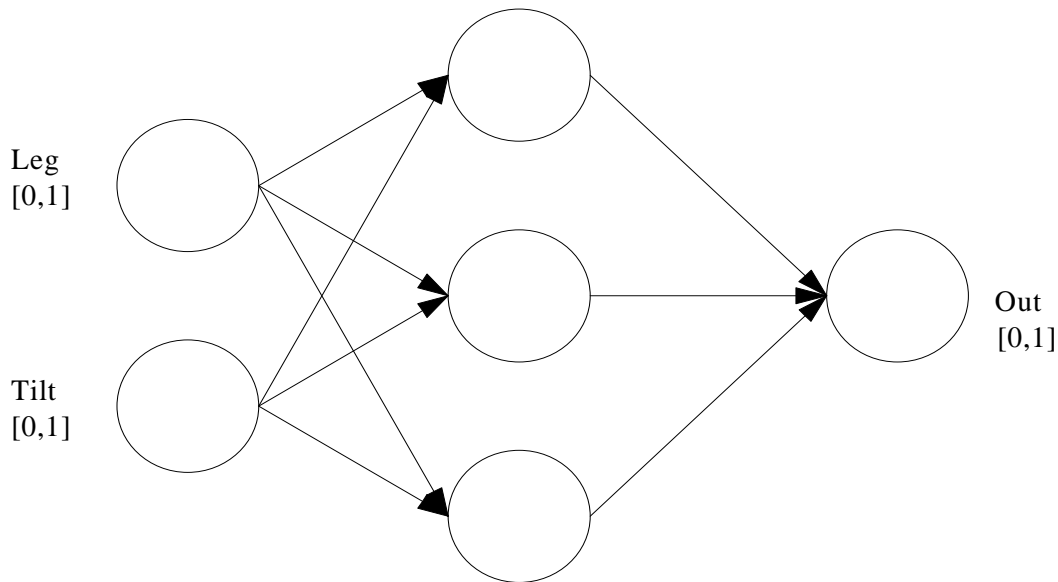
There are many ways to produce training data. The simplest method is to monitor the way a human performs a task and attempt to duplicate the actions of the human. This can cause many problems. First, for situations in which a human can perform the task, they tend to do things too well, and so do not generate a large number of cases in which the system is in a bad position. The result of this is that the system does not learn how to get out of these situations. Secondly, as in the walking gait of a six-legged robot, most humans have no experience in controlling this type of system, and so the data that are obtained from the human are useless. Finally, it requires a large amount of the person's time to create all the necessary examples. This method is best suited to cases in which a human is to be replaced, and so the network can be trained while the human is still performing the task.

In one instance, a six-legged wooden model of a robot was equipped with potentiometers at its joints. This model was then positioned by hand to generate training examples [11]. Pomerleau has put this human training method to good use in the domain of steering an autonomous vehicle. With the addition of a system that generates "imperfect" examples from the more perfect human examples, the ALVINN system can learn to steer by watching a human drive [6].

Simulation is another way to generate training data. This provides the benefit that a human does not have to watch over the data production process at every step; however, it introduces problems of its own. The first problem is that the real world has to be modeled using a computer, which is difficult and expensive to do. There are many factors that are difficult to model, especially those which change through the life of the robot—for example, the wearing of the joints and the variations in performance of the motors.

The method we have chosen to generate the training data is to use the actual hardware. This way no model of the physics of the robot needs to be developed as in simulation. Similarly to simulation, however, the process does not have to be overseen at every step by a human. Basically, a simple algorithm is used to get the desired result for the training data. However, this algorithm is inefficient and does not behave as the resulting, trained neural network should. In our case, this means that it takes several iterations of the algorithm to move the legs to the proper positions—something we do not want to happen in the real robot.

The artificial neural network we use is a multilayer perceptron with 2 input neurons and 1 output neuron (see figure 3.2). There is one hidden layer in the network with 3 neurons. Each



**Figure 3.2:** The architecture of our neural net to level the robot.  
There are 2 input neurons, 3 neurons in 1 hidden layer, and 1 output neuron

neuron also includes a bias input (a connection to a neuron with constant output). The network is trained using a back-propagation-with-momentum algorithm.

### 3.3 Methods (Training the Neural Network)

The robot can be viewed as a platform that can be pitched forward or backward by raising or lowering the front and/or back legs. Using this concept, we then use half of the robot to simulate the terrain's angle and the other half to compensate for that angle. This results in examples for use in training the movements of half of the robot's body. Here is the procedure for generating the training data:

- 1) Randomly select a position for the rear side of the body
- 2) Read tilt sensor and move front legs to move the body towards a level position
- 3) Repeat step 2 until body is level (within a certain allowable error)
- 4) Record the beginning front leg position, beginning tilt, and final front leg position

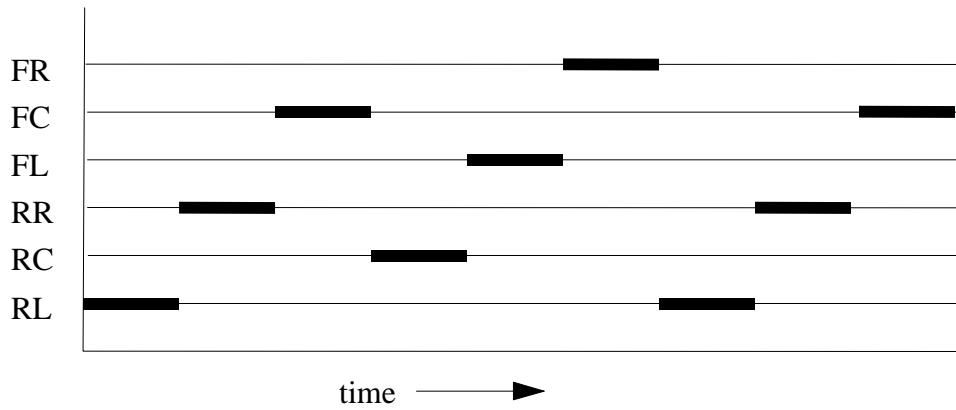
This generates all necessary data to train a neural network with 2 inputs and 1 output. The two inputs are the beginning body tilt and the beginning front leg position. The network is trained to generate the final front leg position necessary to level the robot.

### 3.4 Testing the neural network

The first test performed was very similar to the procedure used to generate the training data. The left half of the body was given a randomly selected vertical position and the network was used to level the body. This test was used to determine that the network was trained correctly and was producing valid results.

The other test performed was a comparison between the walking of Black Widow over a hill simulated with blocks in a static mode and in an adaptive mode using the neural network. In

the static mode all the leg positions for the steps were precomputed as if the robot would be walking on flat land. The walking gait used was one obtained from Beer's book, in which only one leg leaves the ground at a time (see Figure 3.3)[5].



**Figure 3.3:** the dark lines indicate when the specified leg is in the recovery (swing) phase of its motion. The letters at left indicate the leg (Front Right, Front Center, Front Left,...)

The second phase of this test consisted of using the neural network previously trained in conjunction with the static gait. The neural network was allowed to control the down position for the legs on the front half of the body. Any other portion of the motion of the front legs was still controlled by the preprogrammed gait. In the next section we present our results.

## 4.0 Discussion

In this section we will discuss the results of the experiments performed. The first section is on the standing test and the following one discusses the walking tests. The results presented here represent a typical run. The experiments were performed five times.

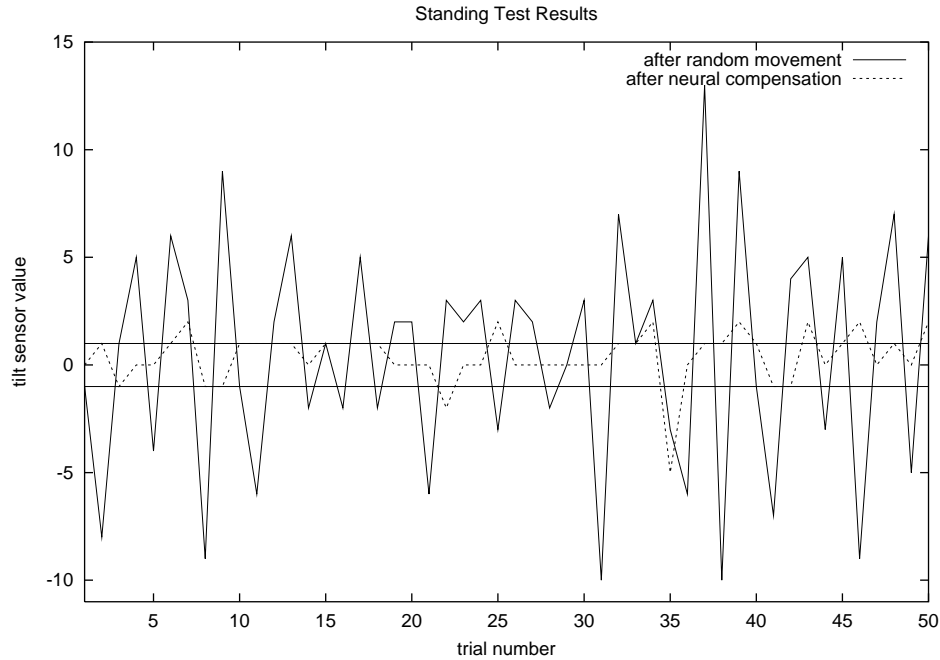
### 4.1 Standing Test Results

In Figure 4.1 the body tilt angles are shown for the different trials<sup>1</sup>. The solid line represents the body tilt after the random leg move was done for the back legs. The dashed line represents the body tilt after the neural network was used to select the position for the front legs. The two horizontal lines show the boundaries of what was acceptable for training data. The training data consisted of examples that brought the body tilt into the range between the two horizontal lines.

Out of the 50 trials performed, 82% were brought to within the desired range. This is good performance especially since those not within the desired range were still close. In none of the examples was the "leveled" robot more than 20 degrees away from being level.

---

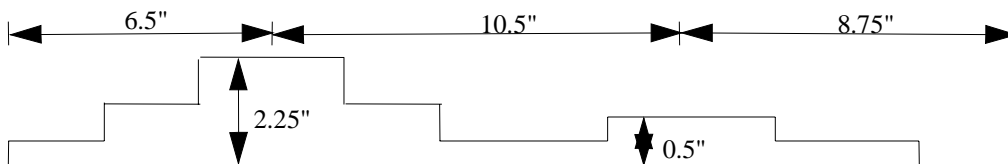
<sup>1</sup> tilt sensor values were translated by -105 to make 0 represent a level body (for all graphs)



**Figure 4.1:** Graph of the standing test results. The tilt sensor value is proportional to the sine of the angle between the robot's body and a plane perpendicular to gravity.

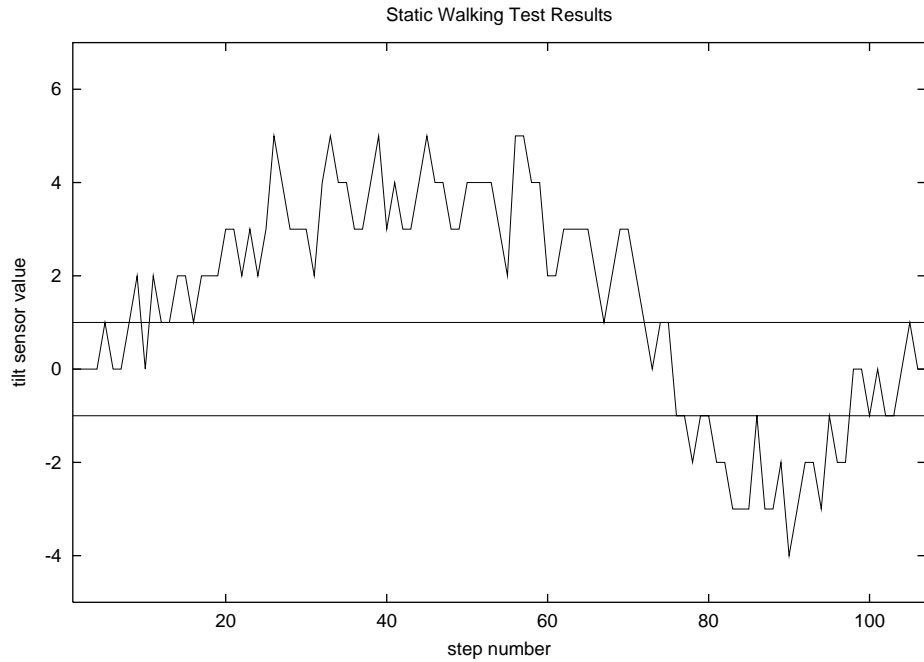
## 4.2 Walking Test

For this test, a short length of simulated terrain was constructed (see Figure 4.2). Black Widow began its traversal of the terrain with all legs on the flat table top to the left of the first hill. The robot then walked over the hill and ended its journey with its legs straddling the second, smaller hill. This test consisted of two stages. In the first stage, the robot was walked over the terrain using a preprogrammed gait that assumed the ground would be flat. The resulting graph

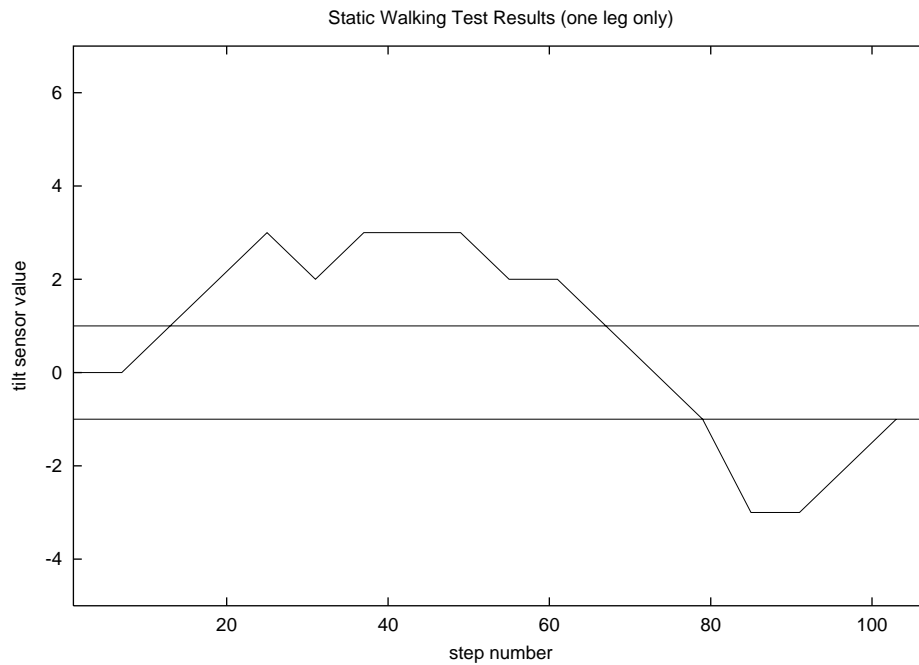


**Figure 4.2:** The hill on which the testing was done

of the body angle can be seen in Figure 4.3. The data at first appear to be quite noisy. However, this is a result of the motion of the robot's body when it moves. When the one leg is picked up for each step of the gait, the entire body leans in the direction of that one leg. If the data are only sampled every 6 steps, the result is a much cleaner graph as can be seen by looking at Figure 4.4. Even though the robot experienced a great deal of difficulty walking over the hill, it still was fairly stable from front to back. However, it experienced trouble with roll in the body. This was partially due to the weight of the tether (it extends from the right side of the body and hangs down). The other factor in this instability is that the legs are much more flexible from side to side. As a result we had to support the robot in some cases to prevent it from toppling sideways.



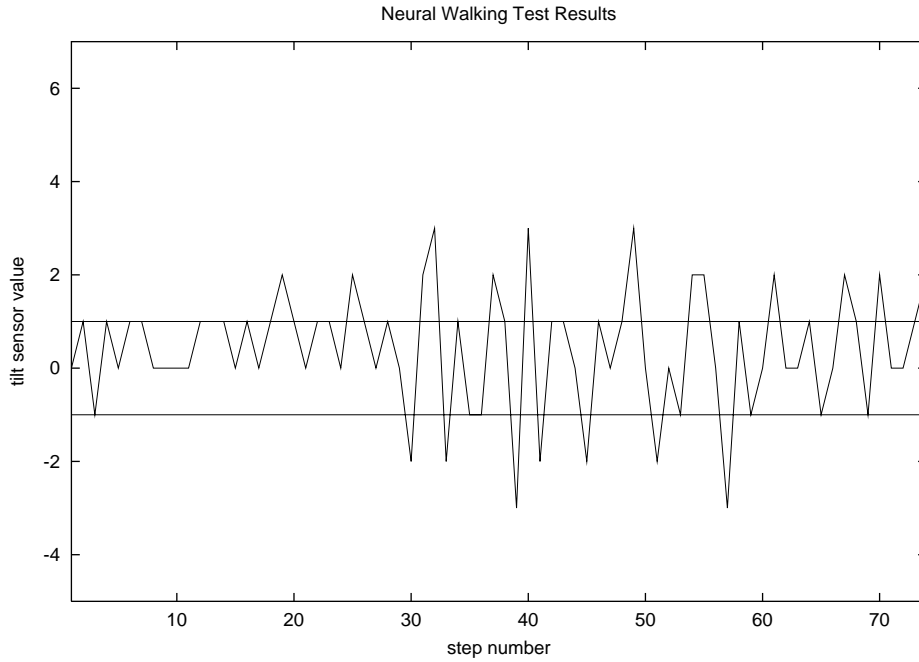
**Figure 4.3:** Results of a static walk over the hill.



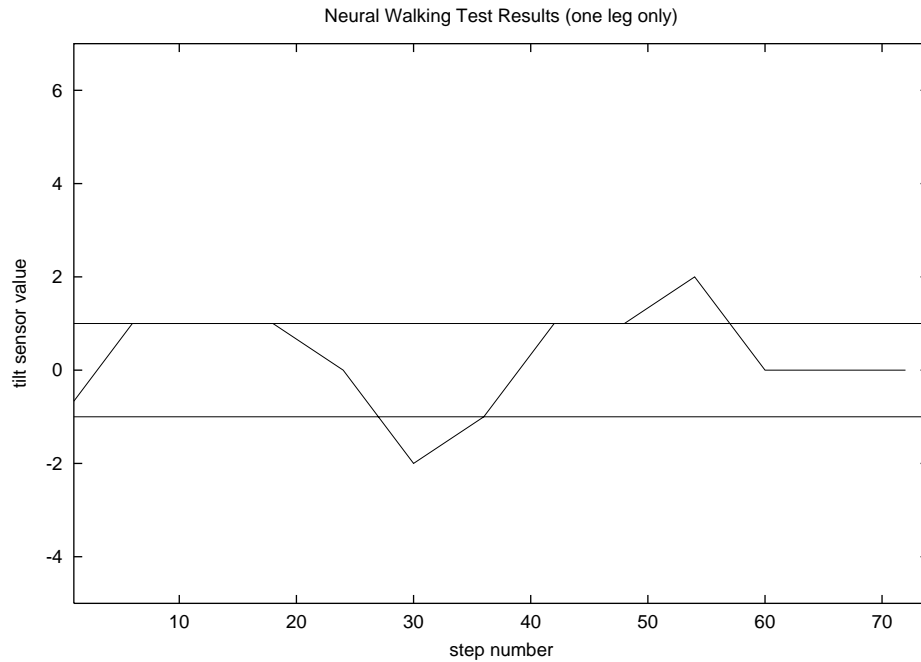
**Figure 4.4:** Tilt levels for static walk over the hill when a specific leg is off the ground. This graph shows how periodic the noise is in Figure 4.2. It is due to which leg is off the ground.

The next run over the hill made use of the neural network. The same walking gait as above was used, except that the neural network was used to determine the "down" position for the front legs. This allowed the neural network to level the body. Figure 4.5 is a graph of the tilt

sensor's values for a trial. The graph exhibits the same "noise" as for the static gait trial and so Figure 4.6 is a graph of the tilt data when the same leg is off the ground (every six steps).



**Figure 4.5:** Tilt Levels for neural network-controlled walk over the hill



**Figure 4.6:** Tilt Sensor Data from the same stage in every stepping cycle (when the same leg is off the ground)

The first thing that was apparent was that the robot had a much easier time climbing over

the hill. Keeping the body level prevented most of the weight of the robot from falling on the downhill legs and so allowed much better movement of those legs. The most obvious result of this is that the robot made it over the hill in fewer steps (74 steps instead of 107). The body roll problems still existed, as we do not compensate for that in the neural network.

As a result of this trial it is obvious that more sensing is needed. The first thing is that a roll sensor is needed to prevent the robot from falling. The other thing we noticed was that foot contact sensing is needed, so that a leg does not hang over empty space, if it reaches its down position and still has not reached the ground yet.

## **5.0 Conclusions and Future Work**

Leveling the body of Black Widow has shown itself to be a valid approach to improve the walking of a legged robot. A result of leveling the body is that the robot's weight is distributed among the legs and so reduces the maximum amount of work a leg needs to do. Not only is a level body useful to walking, but it also has many advantages for sensing and carrying payloads.

The application of neural networks to control of a legged robot has shown itself to be a good idea. The resulting system using the trained neural network to level the body of the robot showed a vast improvement over the static walking method.

Our method of generating training examples has also proven itself to be very useful. In this research we have achieved our goal of automating the parameter adjustments to the neural network. The only purpose the human serves is to transport the data from the generation phase to the training phase.

Future work can build upon this initial successful investigation. The process of generating training data needs to be generalized to the entire body. This requires an extra piece of hardware to change the hill angle as this can no longer be simulated using the body. This could be a platform with two actuators to change the pitch and roll angles of the platform. Using this platform, we would start by standing the robot on the platform and then randomly choosing an orientation for the platform. Then using a similar algorithm to the one shown in section 3.3 we would move the legs on the low and high sides of the body to level it. In this case there would be eight inputs to the network -- the six leg positions and the two tilt sensors. The output would be the six new leg positions. Once again the resulting network would only control the "down" positions of the gait used in the tests.

## References

- [1] Espenschied, Kenneth S. and Quinn, Roger D., "Biologically-Inspired Hexapod Robot Design and Simulation," *Conference on Intelligent Robotics in Field, Factory, Service, and Space (CIRFFSS '94)*.
- [2] Beer, Randall D., Hillel J. Chiel, and Leon S. Sterling, "An Artificial Insect," *American Scientist*, vol. 79, pp. 444-452.
- [3] Chiel, Hillel J. et. al., "Robustness of a Distributed Neural Network Controller for Locomotion in a Hexapod Robot," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 293-303.
- [4] Espenschied, Kenneth S., Roger D. Quinn, Randall D. Beer, and Hillel J. Chiel, "Biologically based distributed control and local reflexes improve rough terrain locomotion in a hexapod robot," *Robotics and Autonomous Systems*, vol. 18, pp. 59-64.
- [5] Beer, Randall D. *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology*. Academic Press, New York, 1993.
- [6] Jochem, Todd and Pomerleau, Dean, "Life in the Fast Lane: the Evolution of an Adaptive Vehicle Control System," *AI Magazine*, vol. 17 no. 2, pp. 11-50.
- [7] Brooks, Rodney A., "A Robot that Walks: Emergent Behaviors from a Carefully Evolved Network," *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*. vol. 2, pp. 692-4, IEEE Computer Society Press: Washington.
- [8] Wettergreen, David and Thorpe, Chuck, "Developing Planning and Reactive Control for a Hexapod Robot," *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pp. 2718-23.
- [9] Bares, John E. and Whittaker, William L., "Configurations of Autonomous Walkers for Extreme Terrain," *International Journal of Robotics Research*, vol. 12, no. 6, pp. 535-59.
- [10] Raibert, Marc H. et. al., *Dynamically Stable Legged Locomotion*, Carnegie Mellon University Technical Report CMU-RI-TR-83-20.
- [11] Berns, Karsten, Rudger Dillman, and Stefan Piekenbrock, "Neural Networks for the Control of a Six-legged Walking Machine," *Robotics and Autonomous Systems*, vol. 14, no. 2-3, pp. 233-44.
- [12] Hertz, John, Anders Krogh, and Richard G. Palmer, *Introduction to the Theory of Neural Computation*, 1991, Addison-Wesley Publishing Company: Reading, MA.