Chapter 6
Initial Results

6.1 Introduction

In this chapter, the initial results obtained from simulating the Direct Growth Method are presented and discussed. Results are presented showing the technique in operation with a simple body form.

6.2 Results from Single Functions

The first problem investigated was the evolution of a Central Pattern Generator (CPG) which could produce the basic gait patterns for bipedal locomotion using the one passive, one active degree of freedom leg with the most basic (mudskipper) body form. Firstly, the MMM neuron model described in Section 5.2 of Chapter 5 was used to implement the CPG. In this case the actuator model is slightly modified so that the leg joint is forced to move up to the knee lock reset point from the forward ground contact point before the robot propels its body forward on the next stride as shown in Figure 6-1.

![Figure 6-1 Modified actuator model](image)

The initial number of neurons in the CPG was set at two because there were two actuators present, each of which must be connected to a neuron. The simulated robot was stable in all directions because it was only the production of the appropriate gait patterns that was under investigation. The fitness score for each chromosome was how far the robot moved from its initial position within 500 time steps (therefore, higher scores were better). Two different modules (firstly, with one neuron and secondly with
two neurons) were added to grow the network, while preserving the neuron parameters and inter-neuron connection weights in the previous modules. All the modules were fully connected. The configuration and growth of the network with two initial neurons proceeded as shown in Figure 6-2. Solid lines show possible connections. The modules were added until the fitness reached its maximum value, and increasing the number of modules thereafter made no difference to the fitness.

Figure 6-2 Growth scheme for single degree of freedom. (a) First module placed and ready to train (b) First module fully trained; second module placed and ready to train

Figure 6-3 shows the resulting robot leg positions, when modules with a single neuron were added to an initial module containing 2 neurons. The best pattern (highest fitness) is when both the legs fluctuate between position 5 and 40, out of phase and the pattern repeats in this range, to give a maximum distance of 430. This corresponds to 14.25 complete strides within the simulation time.

Studying the graph (Figure 6-3(a)), one can see that the left leg is in phase with the right leg at the beginning of the oscillation and the gait pattern stabilizes after this. There were no oscillations in the position (between position 5 and 40) of the robot legs in the beginning when the network size is small but the oscillation becomes clearer in the latter part of the experiment, Figure 6-3(c). The distance moved by the robot with
two neurons in the first module is 341 steps ((d)). The distance remained the same after the second module is added.

Let us consider the operation of the network as more modules are added while freezing the neuron parameters and connection weights of the previous modules. When a new module is added, there are many possible connections between neurons. In this case, for example, when a second module of one neuron is added to an initial module of 2 neurons there are 5 possible connections (including the recurrent connection to itself). More connections are possible as the number of neurons is increased in the module or the number of modules. The solution search space expands as number of connections increases. The larger the search, space the more difficult it becomes for the ES to find a good solution. One of the probable reasons for no increase in fitness is that there were not enough neurons in the new module to influence the previous modules.

After the third module is added the distance increased to 373, an increase of 32 steps. This increase may not be possible without the presence of the second module. The distance remained the same for the next three modules. When the sixth module is added the distance increased by 4 steps and remained the same thereafter with increasing number of modules of one neuron. There is no large increment in distance moved after the third module.

When there is no increment in distance after a new module is added, the previous modules can be said to have reached a stable structure. Most probably, more neurons are required in the new module to modify the initial behavior of the stable structure. In this case, one neuron in a module is not adequate to give a great improvement.

It also can be seen from Figure 6-3(a) to (c) that the leg oscillates between positions 0 and 40, which are not within the desired range. The leg always goes to the 0th position, Figure 6-3(a) - (c), from the rear ground contact point. This means that the distance count loses 5 steps when the leg moves from the rear to the forward position. From Figure 6-3(c), on average there are 12 complete strides between leg position 0 and 40. Therefore the total number of steps was 60 less than the maximum possible. The distance moved by the robot in Figure 6-3(d), increases with increasing number of modules. The maximum distance moved with six modules is 377 steps.
Figure 6-3 Leg positions of a bipedal robot and the improvement of fitness when modules with single neuron were added to the previous modules.
Figure 6-4 shows the leg positions of the robot and the distance moved when modules with 2 neurons were added to the system. In both the legs started to oscillate between position 5 and 45 when there were 4 neurons in total. This behaviour does not occur when modules of one neuron are added to the existing network. The oscillations continue to increase as the number of modules increases. This improves the distance moved by the robot.

The robot moved 358 steps with 2 neurons in the initial module. The rate of change of steps when the second module was introduced was 31. The distance moved increased to 389 steps. The rate of change decreased to 5 and 2 for the third and fourth module. Further changes remains constant at 2. The maximum distance moved was 396.

It can be seen from Figure 6-4 (e), that the distance moved increases with an increasing number of modules, but it is still not possible to reach the theoretical maximum distance. The distance moved increases by 22 steps when the network is grown with a module with 2 neurons compared to when the network is grown with a single neuron module. A good solution was still not achievable by growing the network with 2 neurons in a module.

Even though having 2 neurons or more in the new module may provide more connections, neuron functionality also seems to have an important role in determining the growth of the network. In the MMM neuron model the timing parameters, \( t_1 \) and \( t_2 \) of the neurons are fixed; there is no flexibility to modulate this information. The addition of new modules only provides the required phase shift for a particular gait, in this case bipedal locomotion. This shows that the timing information of the neuron is very important.
Note: x:y:z where x,y,z… refers to number of neurons in a module

Figure 6-4 Leg positions of the robot and the distance evolution when modules with 2 neurons were added to the previous modules

da) 1 module, 2

e) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the 1st module i.e. neuron 1 & 2.

d) 4 modules, 2:2:2:2

c) 3 modules, 2:2:2

Figure 6-4 Leg positions of the robot and the distance evolution when modules with 2 neurons were added to the previous modules
In the next experiments, the MMM neuron model described in Section 5.2 was used to implement the lower layer of the ANS and was tested on an actuator with 2 degrees of freedom as shown in Figure 5-10. However, when this was implemented, it was found that the network failed to evolve to a solution, which moved any distance. The result in Figure 6-5 below shows the robot’s leg positions when 2 and 5 neurons are used in the initial module. The left leg position with five neurons is at position 90; therefore it is not shown clearly on the graph.

This result meant that the system had to be examined to establish why it was failing. It was discovered that this failure was due to the neuron model used.

The above results (Figure 6-5 (a) and (b)) suggested that the MMM neuron model described in Section 5.2 was not capable of producing the required outputs for bipedal locomotion using the 2 active degree of freedom model actuator. This is because the neuron model has a fixed on ($t_1$) and off ($t_2$) time; this causes the neuron to fire for the time fixed by the evolutionary algorithm. The neuron does not therefore reduce or increase its firing rate in response to influences from other neurons. Moreover, in further experiments (below), it was found that influence from other neurons is very important.
Note: \(z:y:z\) where \(x,y,z\ldots\) refers to number of neurons in a module

Figure 6-6 Leg positions of the robot when modules with 1 neuron were added to the previous modules
Figure 6-6 shows the results of using the new neuron model (Spike Accumulation and Delta-Modulation) described in Section 5.2 to evolve a bipedal gait when one neuron is added to the existing modules for the actuator model shown in Figure 5-10 of Chapter 5.

From Figure 6-6(e), the distance moved by the robot increases with increasing number of modules with one neuron. The leg position (Figure 6-6 (a-d)), oscillates between position 0 and 180 without reaching zero like the previous neuron model (Figure 6-3 and Figure 6-4 does with the first actuator model in Section 5.2. This shows that this new neuron model is capable of controlling biped locomotion with these actuators. The distance moved decreases further when a fourth single-neuron module is introduced. There are three possible reasons for the decrement in the distance moved. The first is the inability of the neuron model itself to modulate the firing activity. Secondly, the connection pattern between neurons (within and between newly added modules) is incorrect; in all the experiments described so far, all the neurons in the network were fully connected. Thirdly, when a new module was added to the network, the ES was not able to evolve the best connection weights to increase the distance moved by the robot. Inconsistent activity in the network can cause the decrement in the distance.

Figure 6-7 shows the robot’s leg positions when two neurons are added to the existing modules. From (e), the distance moved by the robot increases for the first two added modules and then decreases for the latter two modules. The robot’s leg position is much improved compared with the single neuron module results. This shows that the number of neurons in a module is very important. Later experiments will give more insight into this point. From Figure 6-6 and Figure 6-7, it may be noticed that the fitness increases quickly at the beginning and then starts decreasing when more modules were introduced.
Note: x:y:z where x,y,z… refers to number of neurons in a module

e) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the 1st module i.e. neuron 1 & 2.

Figure 6-7 Leg positions of the robot when modules with 2 neurons was added to the previous modules
During these tests, a second important discovery was made (the first being the importance of the neural functionality outlined above). This was that allowing all connections to be present - that is, allowing a fully connected network - caused the evolution to either slow down or stop completely. This problem was resolved by allowing the Evolutionary Algorithm to choose the connections within the network as well as their weights. The reason that the connection pattern is important may be that a fully interconnected pattern means that all neurons in the previous module are affected by the new module. While some of these connections cause improvements in fitness, this may be counteracted by other connections which cause a decrease. Although it could be argued that unused connection weights will evolve to zero anyway, it was found that evolution proceeds much more quickly by simply allowing the deletion of connections.

The initial experiments with this approach involved adding a module with one neuron to the previous modules. Figure 6-8 shows the leg positions of the robot for this configuration. The robot managed to move a distance of 261 steps with 2 neurons in the initial module. The distance increased with increasing number of modules and saturated at 310 after the fourth module. The growth strategy of adding a module with one neuron could not evolve fully towards the best solution.
Note: x:y:z where x,y,z... refers to number of neurons in a module

a) 1 module with 2 neurons

b) 2 modules, 2:1

c) 3 modules, 2:1:1

d) 4 modules, 2:1:1:1

e) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the 1st module i.e. neuron 1 & 2.

Figure 6-8 Leg position of the robot when modules of one neuron were added to the network with connections evolved by the ES
The first module used previously to illustrate the growth in adding a module with one neuron was used again in this experiment. Figure 6-9 shows the leg positions when a module with 2 neurons was added to the previous modules. There were an improvement of 89 of steps in distance when the second module was added. The distance continued to increase with an increasing number of modules. The maximum distance moved was 420 steps with six modules. The distance saturated and remained at 420 with increasing number of modules thereafter. There were 12 (six modules of two neurons) neurons in total. Adding 2 neurons in a module showed a great improvement in the results compared to adding a module with one neuron but maximum distance still could not be reached.

A conclusion that can be drawn by analyzing all the results from the previous experiments is that there should be a minimum number of neurons in the new module for it to have a maximum potential for incremental growth towards the best solution. The number of neurons required depends on the mapping difficulties that the new module has to overcome to reach the solution.
Note: x:y:z where x,y,z… refers to number of neurons in a module

a) 1 module with 2 neurons

b) 2 modules, 2:2

c) 3 modules, 2:2:2

d) 4 modules, 2:2:2:2

e) 5 modules, 2:2:2:2:2

f) 6 modules, 2:2:2:2:2:2
e) Distance moved with increasing in number of modules.
Output to the actuator taken from the neurons in the 1st module i.e. neuron 1 & 2.

**Figure 6-9** Leg position of the robot when modules of two neurons were added to the network with connections evolved by the ES

Figure 6-10 illustrates the distance travelled with different numbers of neurons in the modules. The result was promising, and the distance moved and the leg patterns improved as number of modules increased. A module with two neurons was trained. The robot was able to move a maximum distance of 261 in 500 time steps - see Figure 6-10 (a). Then, a module with two neurons was added. The distance moved increased to 350 – see Figure 6-10 (b). Finally, a module with three neurons was added and the distance increased to 440 – see Figure 6-10 (c). The distance moved never changed thereafter, with an increasing number of neurons and modules. Figure 6-10 (d) shows the fitness improvement as modules are added to the network. The total number of neurons to reach the maximum distance for a bipedal locomotion is 7. Figure 6-11 shows the neuron connections between neurons for all three modules.
Note: x:y:z where x,y,z… refers to number of neurons in a module

a) 1 module with 2 neurons

b) 2 modules, 2:2

c) 3 modules, 2:2:3

d) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the 1st module i.e. neuron 1 & 2.

Figure 6-10 Leg position of the robot when variable number of neurons were added to the new modules with connections evolved by the ES
All the neurons in the network are assigned with a numerical Identity (Id) in order of addition to the network. Table 1 below shows the number of modules in the network and the neuron identities in that module. Module 2 to 3 are the new modules evolved on top of the previous modules. Module number 1 is the initial output module.

<table>
<thead>
<tr>
<th>Module Number</th>
<th>Neuron Ids</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1, 2</td>
</tr>
<tr>
<td>2</td>
<td>3, 4</td>
</tr>
<tr>
<td>3</td>
<td>5, 6, 7</td>
</tr>
</tbody>
</table>

Table 1 Module number and neuron Ids

Table 2 shows the evolved connections between neurons when module number 2 and 3 are formed.
By analyzing the connectivity table (Table 2), we can see that there is at least one connection formed from the new module to the output module, shown in bold. It is also noticeable that fewer connections are formed from the new module to previously evolved modules. From Table 2, more connections are formed from the previous modules to the new module, shown in italics.

The important point to note is that, if the evolutionary algorithm does not find a good solution, the synapse weights connecting the new module to the previous modules turn out to be zero. From Figure 6-12 (a) the maximum distance reached was 261. When a new module with 2 neurons was introduced, the initial fitness was preserved for few generations before the distance increased further. This showed that the evolutionary algorithm managed to find that the previous modules (having already acquired some degree of knowledge about the problem) were still able to give the maximum distance, even although the new module made the overall system worse.

A network with 12 neurons was trained and the distance moved was 395. There could be 144 \((12^2)\) connections between neurons if all the neurons are fully connected. A simple mathematic calculation will reveal that there are \(2.23\times10^{43}\) possible network topologies. Since the ES has to find optimal weights for the connections, this indicates that a big ANN is not always the best solution (because of the large search space). The final solution for a problem might be very small in a large space; incremental growth therefore has an advantage under such circumstances.

<table>
<thead>
<tr>
<th>Neuron Id</th>
<th>Connection from Neuron Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2, 4, 5, 7</td>
</tr>
<tr>
<td>2</td>
<td>1, 2, 3, 5, 7</td>
</tr>
<tr>
<td>3</td>
<td>2, 4, 5, 6, 7</td>
</tr>
<tr>
<td>4</td>
<td>1, 3, 4</td>
</tr>
<tr>
<td>5</td>
<td>1, 2, 4, 7</td>
</tr>
<tr>
<td>6</td>
<td>1, 2, 4</td>
</tr>
<tr>
<td>7</td>
<td>1, 2, 3, 4, 5, 6, 7</td>
</tr>
</tbody>
</table>

Table 2 Evolved connections to and from neurons in the network
Note: x:y:z where x,y,z... refers to number of neurons in a module

Figure 6-12 The evolution of distance travelled when variable number of neurons were added to the new modules with connections evolved by the ES

6.3 Quadruped

A network to produce a quadruped trot gait based on the actuator model with 2 active degrees of freedom (Figure 6-13) was evolved. The total number of modules required to produce the gait was 6. The modules contained 5, 3, 2, 4, 4, and 5 neurons respectively. In the previous experiment for bipedal locomotion there were two neurons in the initial module. Each neuron in the module is connected to the first active degree of the actuator. There were 5 neurons in the initial module for this experiment. It was found that having 4 neurons in the initial module did not produce the required phase shift between the legs. Irregularities in the leg position can be seen in the first 3 modules (Figure 6-13 a – c). The leg position stabilised within the desired range thereafter. A total of 23 neurons are required to successfully evolve the trot gait.
to the maximum distance possible. Figure 6-13 shows the leg positions of the robot and distance evolution as new modules are added to previously evolved modules.

Note: x:y:z where x,y,z… refers to number of neurons in a module

a) 1 module with 5 neurons

b) 2 modules 5:3

c) 3 modules 5:3:2

d) 4 modules 5:3:2:4

e) 5 modules 5:3:2:4:4

f) 6 modules 5:3:2:4:4:5
g) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the 1st module i.e. neuron 1 & 2.

![Figure 6-13 Quadruped trot gait leg positions](image)

### 6.4 Permissible Module Connections

Another area addressed in larger networks is that of localising the neural module’s connections. At present, the networks used are small enough to allow any neuron to be connected to any other. However, in large networks, this becomes impractical and smaller connection areas (for example only to the previous module layer) may be required. This type of growth could be called uni-directional because modules are only added in front or at the rear of existing modules.

To analyse the effect of permissible connections in a large network, two different experiments were carried out. In the first experiment, modules are only connected to the rear of the last module. Connections are not allowed between other modules (for example connections between the second and the initial module). The outputs are taken from the initial module. This method is illustrated in Figure 6-14.

![Figure 6-14 Adding modules at the rear of initial module](image)
In the second experiment, modules are added in front of the last module. Again, connections are not allowed between other modules. In this method, the outputs are taken from the newly added module. Any neurons in this module could be selected to be the output neuron. The disadvantage of this method is that there will always be a minimum number of neurons in the module. The number of neurons is determined by the number of actuators. For example, a minimum of 4 neurons are always required in the new module to control a quadruped robot with a single degree of freedom. In the previous method, the number of neurons in the initial module is always fixed. Figure 6-15 illustrates this method.

The actuator model described in Figure 5-8 (section 5-4 of Chapter 5) was used for these experiments. The discussion below starts with the second experiment and then continues with the first.

Figure 6-16 shows the leg positions of the robot and the distance moved when modules with 2 neurons are added in front of the last module. Modules with a minimum of 2 neurons were required to control the bipedal robot because there were 2 actuators (legs with one active degree of freedom). A total of 3 modules with 2 neurons in each was required to produce a bipedal walking gait. The robot managed to move a distance of 240 with 2 neurons in the initial module. It can be seen from Figure 6-16 (a) that the right leg is held at position 10 and the left leg oscillates within the desired range. There is no obvious reason for this output leg pattern. This could be the best solution the ES evolved with 2 neurons in the initial module. Then, a module with two neurons was added. The distance moved increased to 450 – see Figure 6-16 (b).
Next, a module with 2 neurons was added and the distance increased to 480 – see Figure 6-16 (c). This is the maximum distance that the robot could move within the specified time scale. Figure 6-16 (d) shows the distance improvement with increasing number of modules.

Note: x:y:z where x,y,z… refers to number of neurons in a module

a) 1 module with 2 neurons  
b) 2 modules 2:2  
c) 3 modules with 2:2:2  
d) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the new module.

In this type of growth, the previous modules are behaving like an input to the new module. The new module behaves like a new function (F (New)). The previous modules (F (Old,n) where n is the number of previous modules) becomes a subset of the new function (F (New (Old))). This method is very similar to the Tiling Algorithm (as mentioned in Chapter 4). However, in the Tiling Algorithm, all the neurons in the new module are fully connected to the neurons in the previous module. This is not the case with the growth technique presented here.
This method may be not biologically viable, because the connections to the outputs may not always change as new modules are evolved.

The results of the first experiment illustrated above in Figure 6-17 were examined. Figure 6-17 shows the leg positions of the robot and the distance moved when modules of neurons are added at the rear of the last module. In Figure 6-17 (a and b) both the right and leg are nearly identical. Figure 6-17 (d) shows the increment in distance moved with increasing number of modules. The distance moved never changed thereafter, with an increasing number of neurons and modules. There were 2, 2 and 4 neurons in each module.

Note: x:y:z where x,y,z… refers to number of neurons in a module

- a) 1 module with 2 neurons
- b) 2 modules 2:2
- c) 3 modules with 2:2:4
- d) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the initial module.

Figure 6-17 Adding modules at the rear of initial module
The significance of this technique will be become apparent after the addition of the third module. This is because the new (third and $n^{th}$ module, where $n$ is number of modules) modules added after this will have a smaller effect on the previous modules ($n-1$ modules). It is apparent from Figure 6-14 the technique that the newly added module can only affect the previous module. It can be seen from Figure 6-17 (c) that there was a significant improvement in the leg positions when the third module was introduced. The reason for different numbers of neurons in a module has already been discussed in Section 6.2 of this chapter. It was also found that the fitness never increased with increasing number of modules with variable number of neurons thereafter. The maximum possible distance could not be achieved with this type of growth. One possible reason is that there is smaller influence from the newly added module to the earlier modules in the network as more modules are added due to the chain nature of the network structure.

We will now incorporate the second growth technique (Figure 6-15) into the network evolved previously (Figure 6-18). Two modules with 2 and 5 neurons were added to the existing network. It was found that fitness increased with increasing number of modules. The distance moved saturated at 450 steps with despite an increasing number of neurons and modules thereafter.

Figure 6-18 shows the leg positions and distance improvement of the robot for the two newly added modules. Even though the maximum possible distance (480) could not be achieved, the distance travelled was increased by incorporating the first growth method. These results show that bi-directional growth is also an option with large networks.
Note: x:y:z where x,y,z… refers to number of neurons in a module

a) 4 modules with 2:2:4:2
b) 5 modules with 2:2:4:2:5
d) Distance moved with increasing in number of modules. Output to the actuator taken from the neurons in the new module.

Figure 6-18 Adding modules at the front of the last module

6.5 Discussion

In obtaining these results, the objective was to evolve systems which could be compared with previous work done by McMinn [McMinn 2000] [McMinn 2002a].

A total of 7 neurons were required to successfully evolve a bipedal walking gait with the direct growth method (Figure 6-10). The number of generations required to evolve the best bipedal gait was less than 100 (Figure 6-12). It was also found that, when a new module was added, the fitness increased quickly for the first few generations. This shows that the previous modules in the network are contributing to the increment of the fitness. The number of generations was fixed at 50 for every new module added to previously evolved network, unless otherwise mentioned.
[McMinn 2002b] used a more conventional model with a fixed network size and functionality to obtain neural networks capable of both bipedal and four legged gaits. The total number of neurons in the Central Pattern Generator (CPG) used by McMinn was four neurons and these were fully connected (recurrent connections). The final evolved CPG had a lower number of neurons. The suggested number of processing units for the CPG is $2 \times n$ where $n$ is the number of legs (or joints if there are multiple degrees of freedom per leg) based on Golubitsky [Golubitsky 1998]. However, the processing units assumed in the $2 \times n$ suggestion of Golubitsky [Golubitsky 1998] are complex mathematical oscillators, rather than the simple types of neurons as used by McMinn. The Spike Accumulation and Delta-Modulation neuron used in this research is much simpler than the one used by McMinn. McMinn [McMinn 2002b] required 1000 generations to evolve a network to produce a bipedal walking gait. The bipedal walking and jumping gait is the most basic. The number of generations is high because the connection weights and neuron parameters are trained until the best walking pattern is found.

The next gait evolved was the pronk. In this gait all the legs move simultaneously and in phase. The initial set-up of McMinn’s network for quadruped gaits is shown in Figure 6-19. The input to the network was a tonic signal, connected to all the neurons in the network. Four outputs were taken from unique neurons. There was no tonic signal provided to the networks used to produce bipedal (walking, jumping) and quadrupedal (trot, pronk) gait in this research. The network could be said to be self oscillating (generating an output without an input signal).
There were 23 neurons in the network evolved using the growth strategy. The total number of generations required to successfully evolve quadruped trot gait was 104 (see Appendix C, Section C.1). The optimal number of neurons for the same network evolved by McMinn was found to be 16 (rather than the initial setting of 8) which allowed all four legs to be controlled and contributing to the appropriate output patterns. McMinn required 1500 generations [McMinn 2002b] to generate the same gait. The main difference is that this system is open-ended and flexible enough for continued development over and above these simpler systems as will be seen in Chapter 7.

Similarly results for bipedal jumping and quadruped pronk gaits were produced and presented in Appendix C, Section C.2.