Modular Thinking: Evolving Modular Neural Networks for Visual Guidance of Agents

Ehud Schlessinger
Institute of Ophthalmology
University College London
11-43 Bath Street, London
EC1V 9EL, UK
e.schlessinger@ucl.ac.uk

Peter J. Bentley
Dept. of Computer Science
University College London
Malet Place, London
WC1E 6BT, UK
P.Bentley@cs.ucl.ac.uk

R. Beau Lotto
Institute of Ophthalmology
University College London
11-43 Bath Street, London
EC1V 9EL, UK
lotto@ucl.ac.uk

ABSTRACT
This paper investigates whether replacing non-modular artificial neural network brains of visual agents with modular brains improves their ability to solve difficult tasks, specifically, survive in a changing environment. A set of experiments was conducted and confirmed that agents with modular brains are in fact better. Further analysis of the evolved modules characterised their function and determined their mechanism of operation. The results indicate that the greater survival ability is obtained due to functional specialisation of the evolved modules; good agents do well because their modules are more specialised at tasks such as reproduction and consumption. Overall, the more specialised the modules, the fitter the agents.

Categories and Subject Descriptors
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

General Terms
Algorithms

Keywords
Modular Neural Networks, Multi-Agent Systems, Mosaic World

1. INTRODUCTION
It is commonly believed that solving multiple simple subtasks is easier than solving a single complex task. For this reason, decomposing a complex problem into several simpler problems may be a potent way to approach a problem. Unsurprisingly, this seems to be a frequently used strategy in nature as well; it is a well known fact that the human brain works in a modular manner [15]. A neural network that exploits this principle by utilising multiple specialised modules, each trained to solve a specific subtask, should be better than a single large network. Another advantage attributed to this architecture is that it reduces the presence of crosstalk (conflicting training messages that occur as a result of a network trying to learn two tasks [6]).

Several approaches have been taken to achieve this goal. One computational approach literally decomposes the task to several subtasks. This method is referred to as a 'mixtures-of-experts' architecture, and is mostly used for supervised learning tasks [5]. Under this scheme, a gating network is connected to several expert networks. Both the gating network and the expert networks are concurrently trained (or evolved). By the time that the process is complete, the gating network learns to break the task into useful subtasks, enabling each expert network to solve an aspect of the larger task. It has been shown that this approach is faster than using a single network [6]. Interestingly, the way the task is decomposed matters greatly; it has been shown that different ways of decomposing a task affect the quality of the result – some ways are superior to others [1]. Therefore, by understanding how the gating network decomposes the task, interesting insights about the nature of the problem can be gained.

The principles behind 'mixtures-of-experts' can be utilised in many ways. Koza added architecture-altering operators to a genetic programming simulation that were inspired by gene duplication and gene deletion mechanisms [9]. These additions enabled a main program to create subprograms to deal with subtasks dynamically, effectively decomposing a larger problem into smaller subproblems. The effect of adding these operators was an improvement in the performance of the system. The parallels are clear: the main program is equivalent to the gating network, and the subprograms are equivalent to the modules.

Similar principles underlie Brooks' subsumption architecture, which is used for controlling robots [2]. This architecture decomposes the overall robot control task into several simpler behaviourally-oriented subtasks (e.g. avoiding objects, moving, exploring). Each of the subtasks is explicitly solved; combining these solutions enables successful robot control.

Modular neural networks are another type of methods that exploit the same principles. Using this approach, a network that is comprised of several subcomponents is used to solve a task. The structures of these subcomponents can vary greatly, and their number can be dynamic (grow or shrink as needed). Consequently, every component of the neural network has the potential of specialising in a subtask; thus, this architecture is more powerful than a 'standard' architecture. In support of this notion, Rueckl et al demonstrated how a neural network with a modular architecture is better at solving the "what" and "where" vision tasks than a non-modular neural network [12]. In another study, five different architectures were used to control a robot for a garbage collection task [10]. The one that performed best was named 'emergent modular architecture' and had modules competing for control of output units in a control network.

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.

Ensembles are another category of methods. Normally, when evolving neural networks, the best individual is picked from the population. However, there is a lot of information in other members of the population that is not used; ensembles try to use this extra information. An ensemble works by combining outputs of several individuals from the population (using a few different mechanisms), and ideally generalises better than any individual network [16]. A good ensemble is likely to be comprised of individuals that specialise in different aspects of the overall problem, and thus, we consider this method related to the other previously mentioned approaches.

In addition, hybrids of the various approaches mentioned have been created, e.g. a method of evolving a population of modules, which are synthesised into modular neural networks [8].

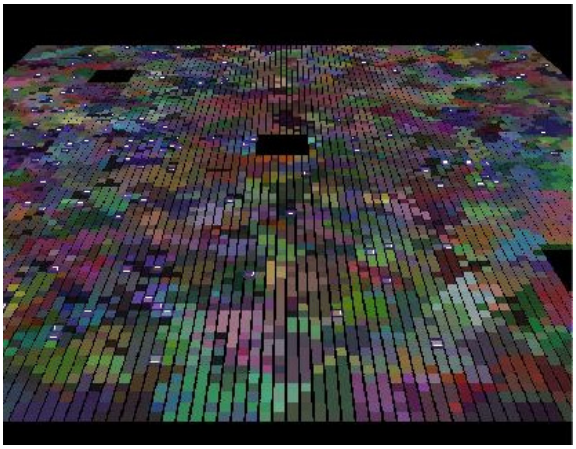

Figure 1. A screenshot of Mosaic World.

In this paper, we expand Mosaic World, an artificial life framework created to understand the computational principles that underlie colour vision [13, 14]. In order to gain useful insights into biological vision, it is crucial that the agents in our system manage to overcome the natural visual challenges Mosaic World presents to them. Thus, gaining an improvement in the quality of evolution in our system is the primary aim of this study. This is accomplished by expanding our existing framework to include a unique hybrid of the above approaches. Additionally, by understanding how and when modular specialisation occurs in our system, valuable insights into agent population dynamics and vision in general may be obtained.

In our system, neural network agents start with a gating network that is linked to one module. The number of modules may grow up to eight, and individual modules as well as the gating network itself, are concurrently evolved. Every aspect of these networks is subject to evolution: number, attributes, topology, weights; allowing our system to evolve both the number and the structure of the modules is a big advantage which most existing systems do not have. This allows evolution to fit the appropriate structure to the subtask, which increases the likelihood it will work well with the overall task [7]. Furthermore, allowing modularity to work at the level of the network is said to have significant advantages [6]. In this sense, we employ a hybrid between 'mixture-of-experts' and modular neural networks.

2. SYSTEM

Mosaic World's environment, a 2D grid of 'coloured' surfaces under multiple simulated light sources, was designed to emulate key characteristics of natural scenes, namely, the fact that objects and illuminants are extended in space. This environment is inhabited by virtual agents, 'critters', that survive by consuming positive resources and avoiding negative resources. Critters consume surfaces which have a positive (food) or a negative (poison) value; the surface's value is determined by its colour. Consumed surfaces slowly regenerate. A surface appears as eaten if 10% or less of its overall value is remaining and as full otherwise.

Critters can reproduce both sexually and asexually. Moreover, there is no explicit fitness function - the critter population is maintained by the critters themselves; if all critters die, a new population is created where 80% are random critters and the rest are mutated clones of critters that showed general promising survival skills earlier in the run. Every critter starts with a certain amount of energy, and slowly loses energy over time. In addition, critters lose energy when moving, turning, consuming resources and reproducing. If a critter runs out of energy, it dies; it also dies if it steps over the edge of the world or falls into one of the holes present in the world. A critter's behaviour is determined by the output of its brain. By evolving the structure of the critter brain, the critter population becomes increasingly better at survival, and inevitably, learns to overcome the visual challenges it faces.

In section 2.1 we describe the standard non-modular critter brains which were used in previous research. In section 2.2 we describe the modular brains used in this study.

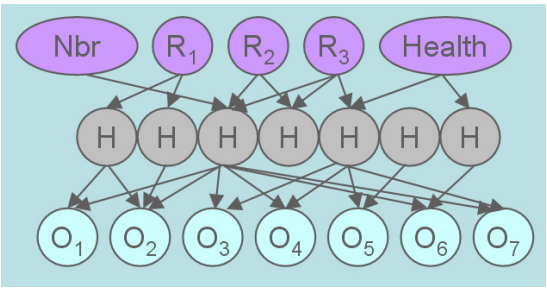

Figure 2. A non-modular critter brain comprised of 3 receptors, the neighbour indicator and health monitor, 7 hidden units. For ease of presentation, this figure disregards the 3D coordinate scheme.

2.1 Standard (Non-Modular) Brains

The critter brain determines the critter's actions. The standard critter brain, which was used in previous research (with minor differences, see [13]), is essentially a modified 3D feed-forward neural network (see fig. 2). It is comprised of multiple 2D layers. The visual layer (effectively the input layer) contains receptors, (input units with novel additions) which receive 'light' stimuli from the environment. The visual layer also has a health monitor unit, which receives the percentage of the critter's remaining health, and a neighbour indicator unit, which receives a signal if another critter is present at the critter's location. The hidden layer contains standard hidden units. The output layer contains output units, which determine the critter's behaviour: turning (left/right), speed of movement (can also be zero), reproduction (sexually or asexually), and resource consumption.

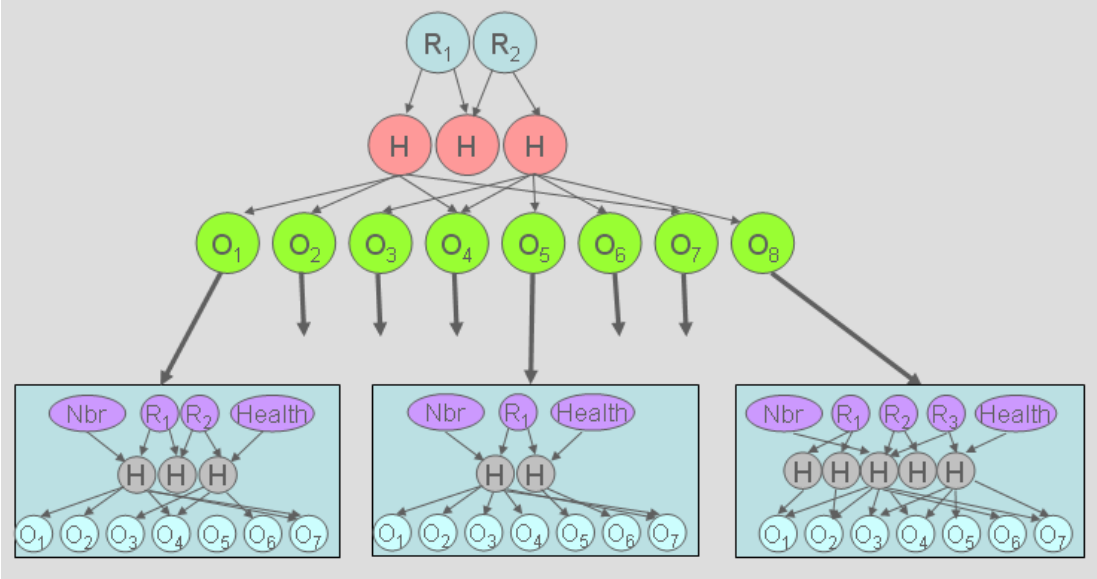

Figure 3. A modular critter brain comprised of a gating network (2 receptors, 3 hidden units), that connects to 3 modules. The first module has 2 receptors, 3 hidden units. The second module has 1 receptor, 2 hidden units. The third module has 3 receptors, 5 hidden units. All modules possess the neighbour indicator and health units. This figure disregards the 3D coordinate scheme.

Every unit in the network has x and y coordinates which define its location in the layer it is placed in. Using these coordinates, it is possible to cross over networks of vastly different topologies during sexual reproduction, as every network possesses the same coordinate reference frame. The receptor's position in the visual layer also determines the location – relative to the critter's centre – where it detects light from (i.e., its visual 'receptive field').

The units of the network communicate through connection weights that extend between units from higher layers to lower layers, and can also connect units to empty coordinates in the network ('partial' connections). Both receptors and connection weights can be either active or inactive. Inactive, 'partial' or 'missing' connections do not participate in the feed-forward process, but are nonetheless inherited by offspring; these are discarded if inactive for a long period of evolutionary time.

2.2 Modular Brains

In previous work we analysed the evolvability of a modular structural mutation operator (adds or deletes complete structures of the network, see [14]). This study showed that although incorporating modular elements into evolution has potential, in our system its disadvantages outweigh its advantages. Based on the premise that modular designs can improve our system, the critter brain has been redesigned with modularity in mind.

The modular visual brain is comprised of a gating network and one to eight modules (see fig. 3). The gating network is almost identical to a standard non-modular critter brain described in the previous section with two exceptions; the first is that it does not have a health monitor unit or a neighbour indicator unit. The second is that it has eight output units. The gating network is evolvable in the same way a standard non-modular brain is (topology, weights, attributes); it receives stimuli from the environment and determines which module to activate at any given time step. The module activated is the one that connects to the output unit that returns the highest activation. When there is only one module, it is automatically active all the time.

The number of modules is evolvable; a critter starts with one module connected to its gating network, and new modules can be added and removed, up to a total of eight modules. Each module is identical to a standard non-modular critter brain in all respects. With this scheme, every action requires two layers of decisions: first, the gating network has to decide which module to activate; then, the activated module determines what action the critter should perform.

It is important to emphasise that only one module can be active at any given moment. Some decisions simply cannot be made by several modules simultaneously: e.g. if there are eight modules and at one point, two modules determine that the critter turn left, two determine that the critter turn right, two determine that the critter move forward, and two determine that the critter stand still - there is no simple way to reach a decision using *all* these choices.

2.3 Genetic Operators

In Mosaic World, mutation takes place during both sexual and asexual reproduction and crossover takes place during sexual reproduction. Crossover of standard, non-modular brains is achieved by 'slicing' all layers of the brains of both critters at a random point. One part is cloned from the first parent and the other is cloned from another. These two parts are combined and form the brain of the offspring. In addition, the new brain is mutated; there are multiple types of mutation operators for standard, non-modular brains: (i) Mutation operators that change receptor attributes. (ii) Mutation operators that change the topology of the network (add unit, delete unit, add weight, delete weight, disable receptor, disable weight). (iii) Mutation operators that change the weight values. (iv) Transmittance mutations that change the critter's colour.

Modular brains use all these mechanisms with the gating network and modules, but have a few more types of mutation operators.

2.3.1 Crossover of Modular Brains

The gating network undergoes crossover the same way a standard non-modular brain does. In addition, a number between 1 and 8 is

randomly picked. All modules from 1 to the picked number are cloned from one parent, and the remainder is cloned from the other parent. If as a result of this process a brain is created with no modules, the process is repeated.

2.3.2 Mutation of Modular Brains

The gating network undergoes mutation the same way a non-modular brain does. The currently active module (the last module that has been active) is mutated normally. The other modules are not mutated (otherwise very quickly they will not be usable at all). In addition, there are four new types of mutations:

'Add Module': the currently active brain is cloned at a given probability (2%). The new module is randomly placed in an empty slot. If there are already eight modules, this mutation is disabled.

'Delete Module': one module is randomly deleted (2%). This module cannot be currently active. If only one module is left, this mutation is disabled.

'Duplicate Module': the active brain is randomly (2% to 35%) duplicated and overwrites the least used module.

'Discard Module': if a module has not been active for a given amount of time steps, it is deleted. The exact number is evolvable.

All percentages were determined empirically, and were modified in experiments described below.

2.4 Module Duplication

In our experiments, we utilised a 'Duplicate Module' operator that acts similarly to gene duplication in nature. We performed multiple experiments with various probabilities assigned to this operator, since the literature stated that this makes a big difference with regards to functional specialisation.

One theory suggests that gene duplication facilitates modular specialisation by first relaxing the selection pressure (as the same module exists twice) and then altering the regulation of this module which causes it to be utilised in a different functional context [3]. This allows the new module to accumulate mutations and specialise [11, 17]. An alternative theory suggests that in nature, the gene being duplicated mostly already has two functions and that gene duplication simply allows the two daughter genes to specialise in one of the original tasks [4]. Indeed, Nolfi demonstrated how hardwired modularity without gene duplication resulted in unspecialised modules in his system [10]. Additionally, Calabretta et al [3] showed that the duplication rate linearly affects performance, with greater values leading to better performance in a robot control task.

3. EXPERIMENTS

Two experiments were performed in this work. In setting the experiments, our goals were threefold. First, we wanted to see whether our novel mechanism would enable critters with modular brains to perform better than critters with the standard, non-modular brains. Second, if this proves to be the case, we wanted to discover how modularity improved the quality of the critters: does specialisation take place, and if so, what sort of specialisation it is. Last, we wanted to investigate how the gating network operated: by discovering when modules are activated, we hoped to gain interesting insights into the problem faced.

To test our hypothesis, we chose a task that critters with non-modular brains had difficulties with: survival in a changing environment. Under this setting, the environment is continually and gradually replaced with a completely different environment. This presents a challenge to the critters not only because they cannot memorise the location of resources, but also because the actual process of changing is greatly 'confusing'; during a significant percentage of the time, there are effectively no good resources, as all surfaces constantly shift. A critter may start consuming a positive resource (a process which takes time), only to find out that the resource became a negative resource by the time it finished. Finally, in order that we determine whether the results depend on the *task itself*, a limited set of runs were performed when the environment is static (non-changing).

The purpose of both experiments was comparing the quality and functionality of critters with modular brains and critters with non-modular brains. Both experiments required a population of evolving critters to be placed in a training world, and ended after 550,000 time steps. Once finished, the critter population was stored and analysed.

The criterion used to measure the quality of the evolved critters was survivability: the average survival ages of critters across runs. Though somewhat arbitrary, this criterion is strongly correlated with fitness as survival requires possessing many important skills (see [13, 14] for a more thorough discussion of critter fitness). This was done by placing 15 copies of the 5 oldest critters of every run in a survival test world (the attributes of which varied depending on the experiment); the critters were expected to survive as long as they could - reproduction was disabled during these runs. The test runs were stopped after 10,000 time steps.

Experiment 1: measuring the quality of evolved runs under a changing environment

In the first experiment, the changing environment operates as follows: after a run starts, every 1300 time steps the world changes *randomly*, a process which takes 300 time steps. This process is repeated until the run ends. Quality is measured using the survival test world, however, unlike training conditions, this environment changes at a faster rate: every 800 time steps there is a 300 time step period of changing. This is done to increase the selection pressure on critters that are not good at dealing with the change period. Six types of runs were conducted; each was repeated at least 16 times:

(1) Critters with non-modular brains: all critters start with the same characteristics (3 receptors, 3 hidden units, fully connected).

(2) Critters with modular brains: all critters start with one gating network (3 receptors, 1 hidden unit, fully connected) and a single module (same structure as the non-modular critters). The probability of the 'Duplicate Module' mutation is 2%.

(3) Same as (2), but the probability of 'Duplicate Module' is 12%.

(4) Same as (2), but the probability of 'Duplicate Module' is 18%.

(5) Same as (2), but the probability of 'Duplicate Module' is 25%.

(6) Same as (2), but the probability of 'Duplicate Module' is 35%.

As mentioned, the reason different mutation probabilities were picked for 'Duplicate Module' was probing the influence this has on the quality of evolved runs and the specialisations that emerge.

Experiment 2: measuring the quality of evolved runs under a static environment

The second experiment investigates whether the results of experiment 1 are different for an easier problem: a *static* environment. This experiment effectively repeats experiment 1, run type 1 (non-modular brains) and run type 5 (modular brains, 'Duplicate Module' value is 25%), but *the environment is static* – it does not change (note: the 'Duplicate Module' value of 25% was used because it was found to produce the most effective results in experiment 1, see results section 4). Quality is measured using the test world. The test world environment does not change.

4. RESULTS

Table 1. Average survival age, average number of total modules, average number of functional modules and percentage of modular critter brains for critters in test worlds; broken down according to category (non-modular critters, modular critters with 2%, 12%, 18%, 25%, 35% probability of 'Duplicate Module')

Category	Average Survival Age	Average # of Modules	Average # of Functional Modules	% of Modules
Non-modular	2341.13	1	1	N/A
Modular (Duplicate Module 2%)	2375.70	3.52	1.58	50.00%
Modular (Duplicate Module 12%)	3021.41	3.98	1.85	68.75%
Modular (Duplicate Module 18%)	2513.76	4.81	1.46	43.75%
Modular (Duplicate Module 25%)	3831.23	4.02	1.78	66.67%
Modular (Duplicate Module 35%)	2051.44	4.85	1.56	38.89%

Table 1, columns 1-2, shows the results of experiment 1 – the average survival ages of the critters, broken down according to category of run. As can be seen, the average survival age for the critters with non-modular brains was lower than the average survival age of most critters with modular brains. Furthermore, the value of the 'Duplicate Module' mutation strongly affected the average survival age: when 'Duplicate Module' was set to 25%, the average survival age was highest (3831.23 – 63% higher than the average survival age of critters with non-modular brains, 2341.13), when set to 35%, the average survival age was lowest (2051.44), in fact, even lower than the average survival age of critters with non-modular brains.

Table 2, rows 1-2, shows the results of experiment 2, the average survival age for critters with modular and non-modular brains in a static environment. Evidently, in this setting too the critters with modular brains survived longer on average than critters with non-modular brains, although the differences were not as extreme.

Table 2. Average survival age for critters with modular and non-modular brains in a static (non-changing) environment

Category of Critter Brain	Average Survival Age
Static environment, non-modular	4279.77
Static environment, modular (all)	5472.04
Static environment, modular (1.5+ modules)	5537.19
Static environment, modular (1 module)	5363.45

5. ANALYSIS

It is clear, then, that incorporating modularity into the brains increased fitness as indicated by critter survivability. However, it is still unknown why this was the case. In addition, it is still unknown whether functional specialisation took place, and if it did, what was the manner of specialisation. To obtain this information, two types of analyses were performed.

For this purpose, it is necessary to know the number of modules: the number of evolved modules is readily available; however, the number of functional modules, modules that *are actually used* is unknown. Thus, the first type of analysis studies the gating network and attempts to discover the number of functional modules and what causes the gating network to activate them. This is obtained by creating five sample worlds and taking 500 random surfaces from each. Every chosen surface is taken five times, in two levels of consumption (9%, 25% left of the surface - these values represent the two states of the surface: 'eaten' and 'full'), and fed to the five oldest critters of all runs. By analysing the activation of the gating network, it is possible to discover the modules that are actually used. Moreover, by examining the stimulus that activated every module, it is possible to understand when the gating network activates the various modules.

Table 1, columns 3-5, shows the average number of modules and functional modules evolved, as well as the percentage of critters with functional modularity (defined as critters that have at least 1.5 functional modules) for critters in the test world in every category of run. In general, the greater the value of the 'Duplicate Module' mutation, the more modules were evolved, though this number does not seem to correlate with the average survival age. However, the number of *functional* modules was correlated with the ability to survive; the longest surviving runs (12%, 25%) had the most functional modules. Interestingly, the percentage and number of functional modules appeared to be influenced by the mutation value – a higher percentage of functional modules was evolved when the mutation value was in a certain range (12% to 25%). This may be the cause of the higher survival age.

By breaking down the results of experiment 1 according to category **and** according to whether functional modularity evolved, a new average survival age figure was created (fig. 4). Evidently, critters with modular brains that utilised modularity were *always* better than critters with modular brains that did not utilise modularity (and used a single module). It is interesting to note that when evolution had the ability of evolving modular brains, and yet did not utilise this mechanism, the results tended to be worse on average than when evolution could not evolve modular brains. Table 2, rows 3-4, shows the results of the same analysis for critters that evolved in a static environment. Under a static environment modularity played a role as well, albeit smaller than under changing environments.

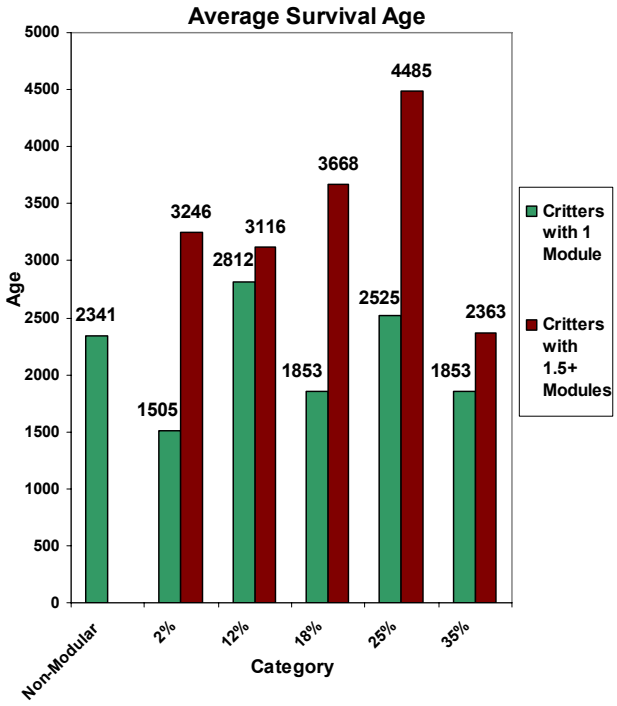

Figure 4. Ave. survival age for critters; break down according to category (non-modular critters, percentage of 'Duplicate Module'), and whether functional modularity evolved.

To understand the differences between critters of varying qualities, several representative critters were picked for the next analyses. Each selected critter was the oldest critter in a run with modular brains that utilised modularity (at least 1.5 modules), thus, the critters used were: 9 very good critters (survived more than 5,100 time steps on average), 6 good critters (survived between 2,100 and 3,300 time steps), and 5 poor critters (survived less than 1,100 time steps).

Table 3. Gating network activation for critters tested; Break down according to critter quality.

Critter Quality	Average Active modules	Module 1 active under all conditions	Module 2 active under all conditions
Very Good	2	33.33%	66.66%
Good	2	33.33%	66.66%
Poor	2	60.00%	40.00%

Table 3 shows the results of the analysis of the gating network activations for the selected critters. First, all critters evolved two functional modules (occasionally there were more, but modules that were utilised less than 2% of the time were disregarded). Second, the operation of the gating network was very consistent: one module was activated under all conditions (surfaces appeared as full or empty), and the second module was only activated when the surface appeared as full. Thus, it appears that one module acts as the primary module (appropriate for all stimuli) and the other acts as the secondary module (used only in some situations).

Although by now we understood when the modules were activated, their manner of usage was still unknown to us. To obtain this information, every representative critter was cloned 75 times and placed in the survival test scenario of experiment 1 for 3,000 time steps. During this time, whenever an action was taken, the identity of the module that activated it was recorded. The actions analysed were: consumption, standing still, slow movement, fast movement, sexual reproduction, asexual reproduction, turn left, turn right (some actions were grouped based on their common properties, e.g. 'turning' refers to both turning left and turning right). In addition, during times the world was changing, it was recorded which modules were active.

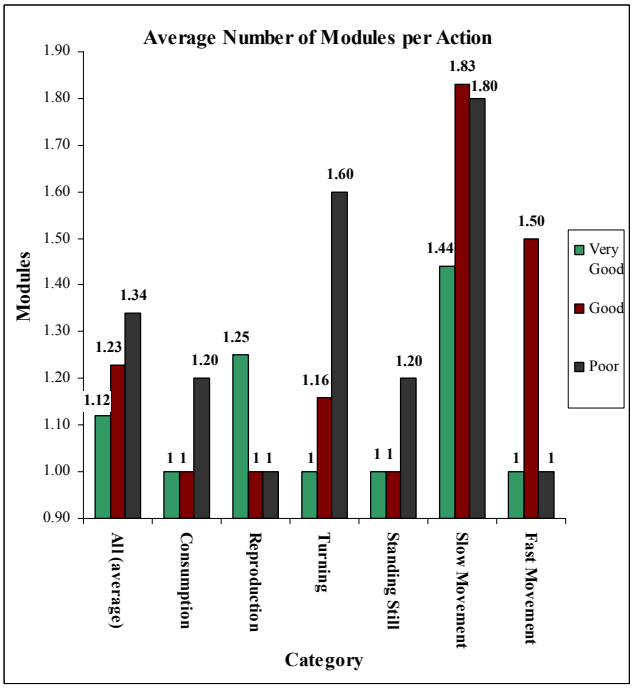

Figure 5. Average number of modules dealing with every type of action; break down according to critter category.

According to the average number of modules assigned to control a given action for every category (fig. 5), the better the critter, the more specialised it was: on average, fewer modules controlled any given action.

When looking at the module analysis of the individual critters, specifically, the division of tasks between the primary and secondary modules (fig. 6), more conclusions are apparent. First, it is clear that the fitter the critter, the more specialised it tended to be: 33.33% of the very good critters were fully specialised, where every action was controlled by a *single* module, contrary to 16.66% of the good critters and none of the poor critters. As for the other critters, almost all were partially specialised, meaning, most actions were controlled by a single module but there were one or more actions which were controlled by both modules; this overlap was often in reproduction or slow movement. Last, 20% of the poor critters (and no critters in the other categories) had no specialisation at all – both modules controlled all actions.

Interestingly, when examining the 'Duplicate Module' value of the runs the critters were taken from, it appears it had no effect on module specialisation. Another surprising discovery was that no specific module was allocated for the times the world was changing - both modules were used regularly during these times.

Critter Quality	#	Specialisation	Actions controlled by primary module						Actions controlled by secondary module					
			Consumption	Reproduction	Turning	Standing Still	Slow Movement	Fast Movement	Consumption	Reproduction	Turning	Standing Still	Slow Movement	Fast Movement
Very good	1	Full	***	***		***	***				***			***
Very good	2	Full			***		***		***			***		
Very good	3	Full	***	***	***	***	***							***
Very good	4	Partial: overlap in reproduction		+	***		***		***	++		***		
Very good	5	Partial: overlap in reproduction		+	***		***		***	++		***		
Very good	6	Partial: overlap in slow movement	***	***	***	***	+						++	
Very good	7	Partial: overlap in slow movement	***	***	***	***	+						+	
Very good	8	Partial: overlap in slow movement	***	***	***	***	++						+	***
Very good	9	Partial: overlap in slow movement	***	***	***	***	+						++	***
Good	1	Partial: overlap in slow & fast movements	***	***		***	+	+			***		++	++
Good	2	Partial: overlap in slow movement	***	***	***	***	++						+	
Good	3	Partial: overlap in slow movement	***	***		***	++				***		+	***
Good	4	Full	***		***	***	***			***				***
Good	5	Partial: overlap in slow & fast movements	***	***	++	***	++	+			+		+	++
Good	6	Partial: overlap in slow movement	***	***	***	***	+						++	
Poor	1	Partial: overlap in slow movement	***	***		***	+				***		++	***
Poor	2	No specialisation - complete overlap	+		+	+	+		++	***	++	++	++	
Poor	3	Partial: overlap in slow movement	***	***		***	+				***		+	***
Poor	4	Partial: overlap in turning and slow movement	***	***	+	***	+				++		++	
Poor	5	Partial: overlap in turning	***	***	++	***	***				+			***

Figure 6: The division of tasks between primary and secondary modules for tested critters. The specialisation column characterises the division of tasks. The two other columns describe which actions are controlled by which module. '*' defines an action that is exclusively controlled by the module. '++' defines an action that is controlled most of the time by the module. '+' defines an action that is only occasionally controlled by the module. An empty space defines an action that the module has no control over.**

6. DISCUSSION

Our results lead to several interesting conclusions. First, enabling the ability to evolve modular brains increased the fitness of the evolved critters on average, a difference that is exaggerated further when only critters that used more than one module are considered. This finding was true for both static and changing environments, though bigger differences were observed in changing environments. Interestingly, critters that did not utilise modularity despite having structurally modular brains occasionally did worse than critters with non-modular brains – this seemed to be linked to the probability of 'Duplicate Module'.

The value of the 'Duplicate Module' mutation had a large effect on evolution of critters with modular brains: too low (2%) or too high (35%), and the overall average survival age was equivalent or worse than the average survival age for critters with non-modular brains. According to the analysis, the modular critters that utilised modularity still did better than the modular critters that did not, however, there appeared to be fewer of the former. In other words, the value of 'Duplicate Module' affected the emergence of functional modules, and consequentially, affected overall critter fitness. Interestingly, once functional specialisation does occur (regardless of the mutation value), it occurs normally – modular specialisation patterns are very consistent. In our view, when the value of 'Duplicate Module' is too low, evolution is unable to successfully utilise the modules for specialisation (as indicated by the gene duplication literature). When it is too high, it becomes disruptive to the evolutionary process. The ideal value is somewhere in between.

Specialisation played a key factor in critter fitness; the more specialised the critters' modules, the fitter the critters were. The very good critters tended to be more specialised, whereas the poor critters often had a large overlap in specialisations, and at times no specialisation at all (complete overlap). As figure 6 shows, the specialisations became less distinct the worse the critters were, and the division of tasks became less logical (e.g. a 'poor' critter #4 divided control of 'turning' and 'slow movement' and as a result did not do very well: if a critter turns while consuming a resource or attempting to reproduce, the action is negated. Therefore, limiting the number of modules that can control this activity appears to be a smart strategy).

Looking at the analysis of the gating network, it is clear the network uses the state of a viewed surface as a cue for switching between modules. Thus, it *always* breaks the stimuli into two groups ('full surface' and 'eaten surface'), and assigns one module to deal with all surfaces (the primary module) and another to deal with only the eaten surfaces (the secondary module). This breakdown is probably why two modules were mostly used - perhaps a different problem may require a different number of modules. When correlating this with the analysis of the modules, two dominant behavioural strategies emerge. **Strategy one:** use the primary module for most actions. When reaching an eaten surface, activate secondary module which specialises in movement and occasionally in turning as well. This strategy works by minimising the danger of consuming a

negative resource; if a surface is already eaten, better to assign a module that can never eat. Furthermore, in some critters this module specialised in fast movement, which can be a very appropriate behaviour when running out of energy and scanning quickly for a good surface. This strategy is very common, and was utilised by most critters. **Strategy two:** use the primary module for movement and turning. When reaching a full surface, activate secondary module which specialises in consumption and reproduction. This strategy works by allowing one module to specialise in tasks that require standing still (consumption and reproduction). Given that identifying the right surface is the most difficult task in Mosaic World (as intended), creating a specialised module for consumption appears to be a good strategy. This strategy was less frequently used and only a third of the very good critters utilised it ('very good' critters #2, #4 and #5 in fig. 6). Interestingly, similar findings were noted for critters that evolved in a static environment, both with regards to the number of functional modules and the behavioural strategies (data not shown).

The above conclusions are consistent with the findings of Calabretta et al [3] which reported that modular networks performed better than non-modular networks. However, in our system, specialisation is the reason behind the increased fitness, whereas in their system specialised modules were not fitter than non-specialised ones. Conversely, in a study by Anderson and Hong [1], modular networks were not fitter than non-modular ones. It is possible that the differences are a result of the nature of the problems that were addressed.

7. CONCLUSIONS

The results confirm the hypothesis that incorporating modularity into brains used for visual agent guidance can greatly improve their capabilities, as critters that evolved modular brains survived much longer on average than critters with non-modular brains. Although critters faced a challenging problem (changing environments), the enhanced survival ability appeared to be derived from the improved capabilities of the modular brains, rather than any specific adaptation to this problem. This improvement in fitness was achieved by dividing the tasks between two modules. Furthermore, the way the tasks were divided was important, the more distinct the division of tasks, the better the overall result. Last, two dominant behavioural strategies emerged. Common allocations of tasks were reproduction and consumption in one module and movement in another.

Although some of these findings are unique to Mosaic World, the conclusions may be relevant to all researchers attempting to use neural networks to solve a task.

8. REFERENCES

[1] Anderson, C. and Hong, Z. (1994). Reinforcement Learning with Modular Neural Networks for Control. *Proc. of NNACIP'94, the IEEE International Workshop on Neural Networks Applied to Control and Image Processing*

[2] Brooks, R. A. (1986). A Robust Layered Control System For A Mobile Robot, *IEEE Journal of Robotics And Automation*, RA-2, April. pp. 14-23.

[3] Calabretta, R., Nolfi, S., Parisi, D. and Wagner, G.P. (2000). Duplication of modules facilitates the evolution of functional specialization. *Artificial Life VI*: 69—84.

[4] Hughes, A. L. (1994). The evolution of functionally novel proteins after gene duplication. *Proc. of Royal Society. Series B* 256:119-124

[5] Jacobs, R.A., Jordan, M. I., Nowlan, S. and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3:79–87.

[6] Jacobs, R.A., Jordan, M.I. and Barto, G.A. (1991) Task Decomposition Through Competition in a Modular Connectionist Architecture: The What and Where Vision Tasks. *Cognitive Science* 15(2): 219-250

[7] Jacobs, R.A. (1997) Nature, nurture, and the development of functional specializations: a computational approach. *Psychonomic Bult*, Rev. 4, 299-309

[8] Khare, V. R., Yao, X., Sendhoff, B., Jin, Y. and Wersing, H. (2005) Co-evolutionary Modular Neural Networks for Automatic Problem Decomposition, *Proc. of The 2005 IEEE Congress on Evolutionary Computation* , Edinburgh, Scotland, UK, September 2nd--5th 2005, 2691--2698

[9] Koza, J.R. (1995). Gene duplication to enable genetic programming to concurrently evolve both the architecture and work-performing steps of a computer program. *Proc. of the 14th International Joint Conference on Artificial Intelligence*. San Francisco, CA: Morgan Kaufmann. Pages 695–717.

[10] Nolfi, S. (1997). Using emergent modularity to develop control systems for mobile robots. *Connection Science*, (10) 3-4: 167-183.

[11] Ohno, S. (1970). *Evolution by Gene Duplication*. Springer-Verlag, New York.

[12] Rueckl, J.G., Cave, K.R. and Kosslyn, S.M. (1989). Why are "What" and "Where" Processed by Separate Cortical Visual Systems? A Computational Investigation. *Journal of Cognitive Neuroscience*, 1, 171-186.

[13] Schlessinger, E., Bentley, P. J. and Lotto, R. B. (2005) Evolving Visually Guided Agents in an Ambiguous Virtual World. *Proc. of Genetic and Evolutionary Computation Conference (GECCO 2005)*, June 25-29, 2005, Washington, D.C.

[14] Schlessinger, E., Bentley, P. J. and Lotto, R. B. (2005) Analysing the Evolvability of Neural Network Agents through Structural Mutations. *Proc. of European Conference on Artificial Life (ECAL 2005)*, September 5-9, 2005, Canterbury, UK

[15] Shallice, T. (1988). *From Neuropsychology to Mental Structure*. Cambridge: Cambridge University Press.

[16] Yao, X. and Liu, Y. (1996), Ensemble Structure of Evolutionary Artificial Neural Networks, *Proc. of the Third IEEE International Conference on Evolutionary Computation (ICEC'96)*, Nagoya, Japan, May 20-22 1996, pp.659-664

[17] Zhang, J. (2003), Evolution by gene duplication: an update, *Trends in Ecology and Evolution*, 18:292-298.