Minds for Robots

Christopher MacLeod and Grant M Maxwell

In the 1950s and 60s, popular culture was entranced by robots. There was Robby in "Forbidden Planet," Gort in "The day the Earth stood still" and many others. This fascination has continued to the present day, only the names have changed, now we have Commander Data of "Startrek" and the NS-5 of "I Robot." Yet despite the interest and the obvious advantages of having intelligent machines to do dirty, boring or dangerous jobs, we are little nearer to creating a similar technology in the real world. This article outlines a new route towards such intelligence in machines - Incremental Evolution.

Approaches to AI

Historically there have been three approaches to creating intelligent machines. The first approach was to try and copy the biological brain. However, this contains billions of nerve cells (also called neurons), interlinked by trillions of connections and unravelling it is well beyond our current scientific capabilities.

The second approach, known as symbolic AI, was to build a large database of rules and instructions to allow the machine to make complex decisions. But we know that biological brains don't work like this. Simple rules also don't have the flexibility to work in real-world situations and in any case, the database soon grows to unmanageable proportions.

Finally, there is the approach of modelling simple neural circuitry using "Artificial Neural Networks" - networks of simple processors, loosely based on real nerve cells. But the resulting circuits are not complex or flexible enough to allow intelligence to emerge.

Artificial Evolution

The problems outlined in the paragraphs above led researchers to consider the mechanism which allowed nature to design the biological brain - Evolution. Evolution works because the fittest individuals in a society tend to survive better and therefore have a higher chance of breeding and passing on their traits to the next generation. Over many generations, therefore, the fitness of the whole population increases.

Engineers have designed artificial versions of evolution called Evolutionary Algorithms. We can illustrate how these work using an example. Suppose we want to design a filter as shown in fig 1.

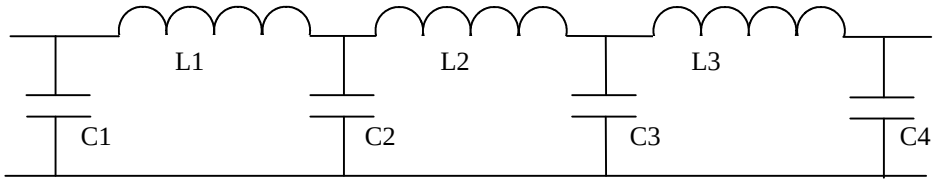
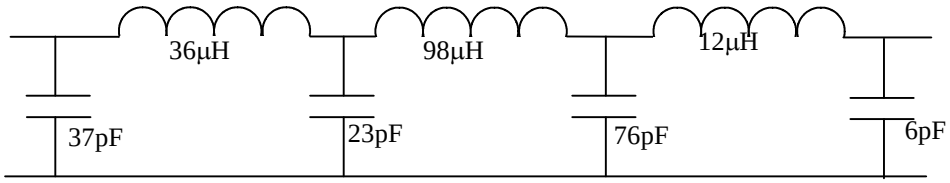


Fig 1. A simple filter circuit.

The circuit can be represented by a *string* of numbers which are its component values. Fig 2 shows a circuit with its components and the string which represents it.



String or Chromosome representing circuit

37	36	23	98	76	12	6

Fig 2. An particular filter and its representative string.

We can generate lots of such strings (also sometimes called *chromosomes*) representing circuits like this - each chromosome is the component values of a single circuit. The starting point of an Evolutionary Algorithm is a number of such strings (called the *population*) filled with random numbers - that is, with random component values.

We then test each of these random strings to see how well it performs - in this case by comparing how good it is with the perfect response. Having done this, we get rid of the worst strings - if we have a population of 99 strings say, we'd delete perhaps two thirds of them - the two thirds which are furthest from the ideal response. This would leave us with 33 strings. What we do next is breed them - just as in a population of animals, the fittest survive and breed together. The breeding process is called *crossover* or *recombination*. It works like this: take two random strings and select a random point on them, then cross over all the numbers to the right of that point into the opposite string and vice versa, fig 3 shows the idea.

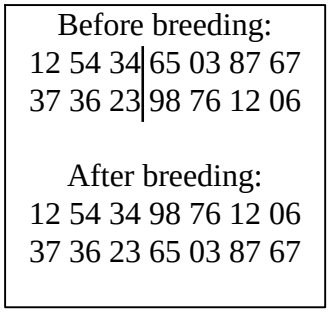


Fig 3. Crossover, also called Recombination

Hopefully, in some of these events, the new string will inherit some good traits from the mother string and some different ones from the father - and will therefore be better than both. Once this is done, the original population of 99 is recreated (the 66 new strings being the result of breeding). A small percentage of the numbers are then changed at random to add some variation into the population - an operation called *mutation* and we start the whole process all over again - evaluation, culling, breeding and mutation. Over many generations of doing this, the population will improve to the point where some of them can solve the design problem satisfactorily. The technique is particularly useful in circuits which don't have good design rules or where compromises have to be made in the design.

Problems

The procedure outlined above has been shown to work well in many problems. It has been used to design Analogue and Digital Circuits as well as DSP systems and Artificial Neural Networks.

However, when we try and use it to design circuits to control intelligent machines, it has only limited success. The main reason for this is that the algorithm is inefficient when applied to large circuits. This is because of the many different combinations of components and wiring possible in such circuits - we say that the circuit has a large *search space*. So what is successful for small problems is unworkable for large ones.

A related problem is that it is difficult to design the algorithm so that it can evolve a circuit which deals with many different kinds of input and output. For example, in a robot, there may be inputs from vision sensors, bump sensors and audio sensors. If you try and integrate these together, they sometimes conflict and interfere. This is why the human brain is not one big network, but lots of little ones (sometimes called modules), all doing their own thing but co-operating together.

How nature does it

Although current Evolutionary Algorithms model the selection of good traits within a population, this is only one feature of natural evolution. If we look at the evolution of animals over time, the other important aspect is *growth in complexity* - animals have changed from simple to complex forms over time, driven by competition.

When life started on Earth it was very simple - just single cells in fact. From then until now, it has slowly but surely become more complex. Single celled forms were

replaced by simple multicellular animals, which eventually developed into jellyfish, then worms, complex invertebrates, fish, amphibians, reptiles and finally mammals - a march of progress from simple to complex. The extraordinary animals of today are only possible because they built on the successes of the past - adding more functionality to those which had gone before.

So, it is not just about blind growth - but about building upon a previously laid, firm foundation. The past triumphs of evolution are not discarded by nature, but new developments are built on top. This is particularly obvious in the brain, were it would have been impossible to rewire billions of neurons and trillions of connections each time it developed - the new changes had to build on top of the older ones. In fact, Richard Dawkins, in his famous book on evolution: “The Blind Watchmaker,” points out that incremental change is the only realistic way to evolve complex animals.

In other words, a traditional Evolutionary Algorithm simulates the selection of good traits in a population of (say) amoebas, but the amoebas can never evolve into jellyfish because the strings which represent the population members cannot become larger and more complex - and even if they did, the algorithm could not handle the extra complexity of the search-space generated.

Incremental Evolution

Incremental Evolution adds the idea of growth in complexity into Evolutionary Algorithms. We can illustrate how it works by considering the evolution of a circuit.

The algorithm works by initially making the circuit as simple as possible. An ordinary Evolutionary Algorithm is then used to evolve this simple system until it performs its task perfectly (or cannot get any better). This simple evolved circuit is then fixed and not allowed to change. Next, a *module* of new components is added to the circuit; these are added to the pre-existing structure and are allowed to evolve. Again, when they can’t get any better, the whole is fixed and another new module is added. Fig 4 shows this process operating on a filter.

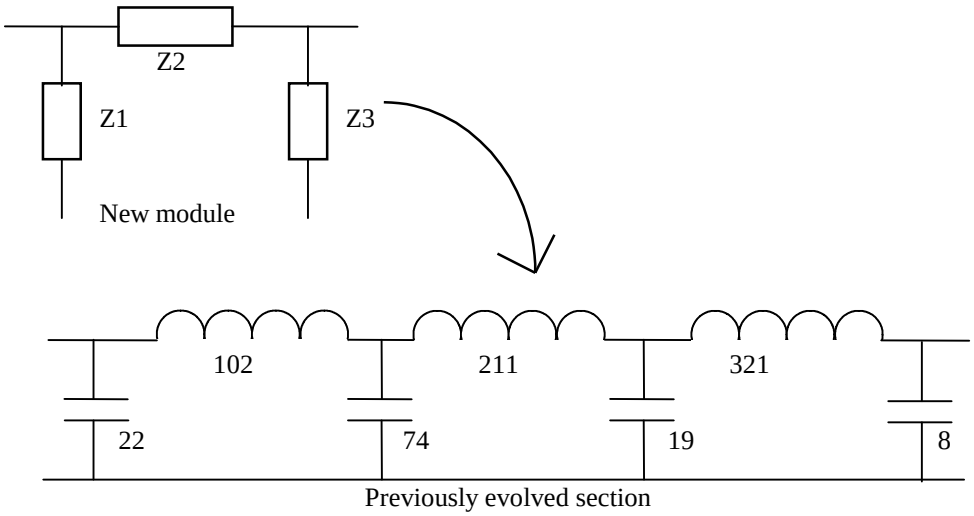


Fig 4, A new module being added to a pre-evolved section.

A better understanding of this can be had by considering the evolution of a radio circuit. If we were to take dozens of transistors and other components and use a traditional Evolutionary Algorithm to design a radio receiver, it wouldn't get very far; the circuit is too complex - the search space too large. However, if we only allowed one transistor and a few components, chances are that it would evolve a simple detector circuit, rather like a crystal set. Now, this doesn't work very well, but it does demodulate signals. If we now fixed this and allowed the algorithm to add a module, we have a good chance of evolving an RF amplifier in front of the detector. Fix all this and we might get an AF amplifier behind it. Try again and you might get a selection filter to tune it at the very front.

So, by allowing the system to grow like this, we can evolve much more complex systems than a simple Evolutionary Algorithm can - and systems which are automatically modular; for example, the filter in fig 4 can evolve to give a much more complex multi-stage response than is likely to be produced by a single stage of evolution. In fact, it turns out that in large systems with many connections, the search space of a traditional Evolutionary Algorithm grows frightenly quickly, whereas in Incremental Evolution, the search space is simply that of the added module.

Robots

In our lab at the Robert Gordon University in Aberdeen, Incremental Evolution was used to evolve a legged robot with vision sensors. The robot's sequence of evolution is shown in fig 5.

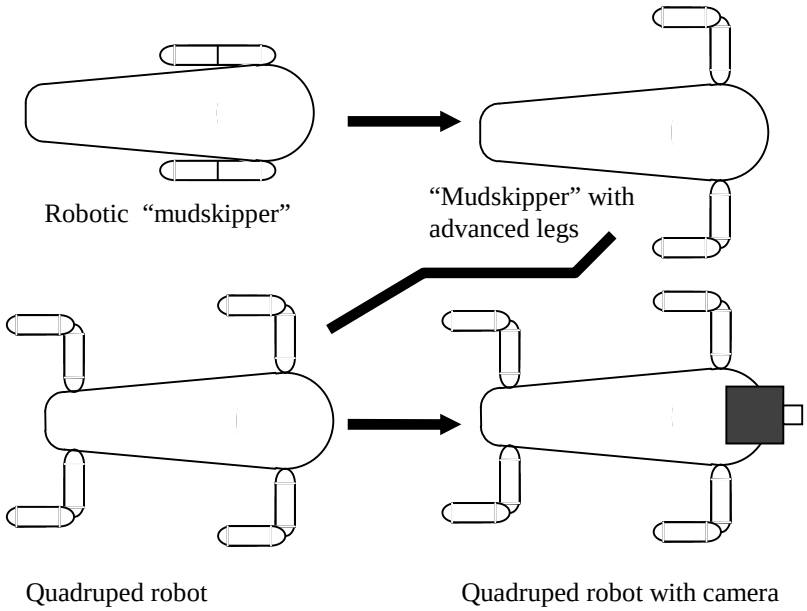


Fig 5, An evolving legged robot.

An important principle can be seen in operation here. The robot's "body" and "mind" must evolve together - the principle of Incremental Evolution applies to the whole system, not just the circuit which controls it. This again keeps the search space in each iteration small - you might call it a holistic form of Artificial Evolution. After all, in nature, the brain of jellyfish goes with its particular simple body plan. In the terms of

Evolutionary Algorithms, we say that the “fitness function” changes along with the current evolutionary stage.

The robot started of as simple as possible; in this case as a robotic “mudskipper.” All it was able to do was pull itself along the floor with two simple inflexible legs. A neural network controller (basically a control circuit) was evolved to allow it do this. Next, the robot was given more complex legs - with two active degrees of freedom (a “hip” and a “knee”). Keeping the previously evolved network, another was evolved to control these new functions - the old network controlled the “hip” and the new, added one, the “knee.” Having done this, another two legs were then added, making the robot into a quadruped; again, what was previously evolved was kept and further modules added on top of the old to control the new legs (although the algorithm was allowed to “cut and paste” previously evolved modules into new positions).

Having got our walking four legged robot, a vision system was evolved for it. This started off as just as one pixel and was allowed to evolve slowly - using the principles above, into a five by five pixel “robotic eye” which could recognise simple patterns as shown in fig 6.

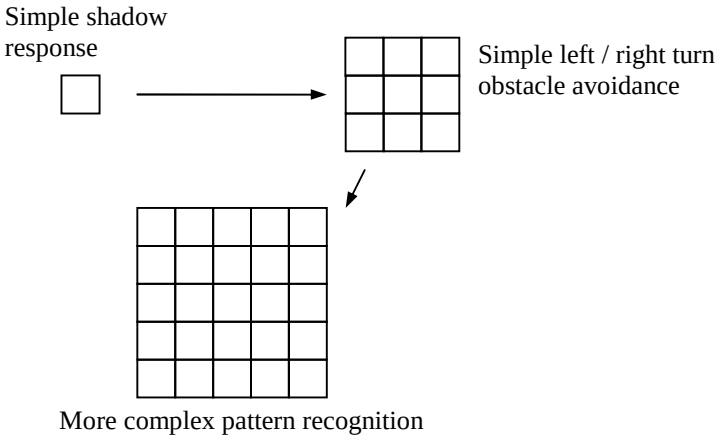


Fig 6, The development of a robotic eye.

New networks were then added to interface those of the legs with the eye and produce a robot which could recognise patterns, turn and alter its gait accordingly. The final robot had over two hundred neurons in twenty two modules and could operate successfully in a variety of environments. Fig 7 shows how one particular function evolved (in this case, the controller for a leg) as new modules were added.

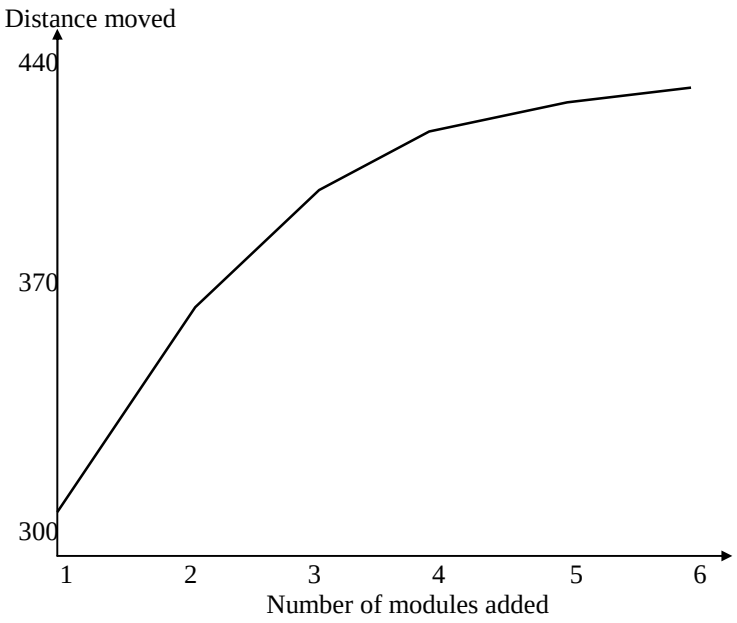


Fig 7, A typical graph showing performance improvement as new modules are added.

The result of all this, is that, not only can a much more complex system be evolved, but the result is modular with parts of the network handling different functions. Hence, it has the ability to connect and integrate different sensor and actuator functions together; this modular network, however, is part of a fully functioning and co-operating whole.

Rules of Evolution

During the process of evolving the robot, it was discovered that there are some important "ground rules" for making the method work. If these are ignored then the evolution will stall and the system won't reach its full potential.

Firstly, the components in each module added must be as flexible as possible - in the case of electronics, the algorithm should be allowed to choose these to be any component (resistor, capacitor, inductor, etc). In the case of a neural network, the flexibility of the neurons must be as high as we can make them.

Secondly, the algorithm evolving the system must be able to choose which connections are made as well as the component values and types. It was found that if this wasn't allowed to happen, the connections interfered with each other and stopped the system reaching its full potential.

Finally, there is usually a minimum number of components needed in each new module, if there are not enough, then the system won't evolve correctly. This is similar to the first and second points - they all indicate that the functionality of the added modules is critical and that each module must be as flexible as it can be.

It's also worth noting that even when the rules are followed, it is possible for evolutionary "dead ends" to be reached, where further improvement is not possible - this also happens in the world of nature. In this case, modules must be removed and the system allowed to re-evolve in a different direction.

Applications

There are also many possible applications of Incremental Evolution in other areas of engineering. One of the most interesting is in biomedical engineering, in the development of prosthetic limbs. In this case, there is a direct incremental route from a simple system with all the joints of the limb fixed in place, to a complex system with all the joints free and controlled. As each joint is freed the controlling circuitry can be allowed to develop by adding to its structure, in the same way as in the robot. Other examples include mechanical engineering - for example in aerospace engineering; here too it is fairly obvious that sections of an aerodynamic design can be added incrementally. There may even be lessons which biologists can learn about the early development of life and what works and doesn't in terms of evolutionary mechanisms.

Today we try and design complex systems inspired by biology, like intelligent robots, by analysing how humans and animals work and copying them. But when we see a human, we see four and a half billion years of evolutionary progress. Each stage of that journey was built on the previous stage. Humankind did not come into being suddenly, without warning. They began their long trek as a single celled organism in the primeval seas. We, our behaviour and our physical makeup, are a result of that development.

What, therefore, makes us think that we can build sophisticated intelligent systems like robots from scratch? Surely they too must build up their bodies and their behaviour in a similar way, just as we did, piece by piece, one step at a time.

References:

http://www.rgu.ac.uk/eng/compint/page.cfm?pge=12909

S Muthuraman, C MacLeod and G Maxwell, The development of Modular Evolutionary Artificial Neural Networks for Quadruped Locomotion, IASTED ACS 03. p268 - 273

S Muthuraman, G Maxwell, C MacLeod, The Evolution of Modular Artificial Neural Networks for Legged Robot Control, Artificial Neural Networks and Neural Information Processing, Springer (LNCS 2714), 2003, p488 - 495.

Sethuraman Muthuraman, "The Evolution of Modular Artificial Neural Networks", PhD Thesis, The Robert Gordon University, 2005

C. MacLeod, G. M. Maxwell, S. Muthuraman, Incremental Growth in Modular Neural Networks, Engineering Application of Artificial Intelligence, Vol 22, Issue 4/5, 2009. pp 600 – 666, doi:10.1016/j.engappai.2008.11.002.

MacLeod C, Practical Algorithms for Incremental Growth, The School of Engineering, The Robert Gordon University, Aberdeen, 2010.