Book Title
Book Editors
IOS Press, 2003

Tactical modularity for evolutionary animats

Ricardo A. Téllez and Cecilio Angulo

rtellez@lsi.upc.edu, cecilio.angulo@upc.edu

Technical University of Catalonia, Jordi Girona, 1-3, Barcelona

AbstractIn this paper, we use a massive modular architecture for the generation of complex behaviours in complex robots within the evolutionary robotics framework. We define two different ways of introducing modularity in neural controllers using evolutionary techniques, which we call strategic and tactical modularity, and show at what modular levels each one acts and how can they be combined for the generation of a completely modular controller for a neural networks based animat. Implementation results are presented for the garbage collector problem using a khepera robot and compared with previous results from other researchers.

Keywords. autonomous robot control, neural networks, modularity, evolutionary robotics

1. Introduction

In the framework of evolutionary robotics, the generation of complex behaviours in complex robots is still an open question far from being completely solved. Eventhough lately some complex behaviours have been implemented in simple wheeled robots [13], it is not clear how those implementations will scale up in more complex robots, understanding by complex robots, robots with a large number of sensors and actuators that must coordinate in order to accomplish a task or generate a behaviour. In our research we focus on this problem by using evolutionary techniques for the generation of complex behaviours in complex robots.

We think that the solution to this problem must be grounded in the divide and conquer principle, this means, the use and generation of modular controllers. This is not indeed a new approach and has been adopted by other researchers [5][10][4][2]. However, in all those modular implementations, modules were created at the level of behaviours, it is, a module was created for each behaviour required by the robot.

In this paper, we introduce the idea of modularisation at the level of robot device and promote its use together with the modularisation at the level of behaviours. We propose that the use of both types of modularisation in a unique controller may be required in order to achieve complex behaviours in complex robots. By doing this, a modularisation at the two levels may allow the generation of really complex behaviours.

This paper is divided into the following sections: section 2 makes a review of related works, section 3 describes the two different types of modularisation approaches; section

4 describes the experimental framework used; section 5 describes the experiments performed and their results; section 6 discusses the results obtained; and final section 7 goes to the conclusions and points towards future work.

2. Modularity in robot control

In the design of modular neural controllers, most of the jobs have been influenced by Jacobs et al. mixture of experts [8]. Their basic idea was to create a system composed of several networks, each one in charge of handling a subset of the complete set of cases required to solve the problem. Their system has been widely used and improved upon by several authors [3][1] in different classification problems. In evolutionary robotics, Tani and Nolfi [16] improved Jacobs' architecture by drawing on a mixture of experts whose segmentation of competences was produced by observing the changes in the dynamical structure of the sensory-motor flow. In a more recent work, Paine and Tani [14] studied how a hierarchy of neural modules could be generated automatically, and Lara et al. [9] separately evolved two controllers that performed two different behaviours, then evolved the connections between them in order to generate a single controller capable of performing both behaviours.

On a similar way, Nolfi [12] makes one of the first attempts to use modular neural controllers for the control of an autonomous robot in a quite complex problem. He compares the performances of different types of control architectures, including two modular ones: on the first one, the controller is composed of two modules each one in charge of one specific and well differentiated task. The division in those two modules is performed by using a distal description of the behaviours, it is, a division in behaviours described from the point of view of an external observer. In the second modular architecture, Nolfi proposes a control architecture where modules are created based on a proximal description of behaviours, it is, the description of the behaviour from the sensory-motor system point of view of the agent. The work of Nolfi was later improved in [15] by showing how a distal modularity could arise from an evolutionary process, and extended by Ziemke et al. [20] by using other types of neural control architectures.

Except Nolfi's proximal division, all the works shown have in mind the implementation of behaviours as composed of sub-behaviours. Hence, the idea in all of them is almost the same: to divide the global behaviour into a set of simpler sub-behaviours, which when combined, make the robot perform the required global behaviour. This division is sometimes made by hand or by any automatic way. Then, each sub-behaviour is implemented by a single monolithic neural controller that takes as inputs the sensor values, and as outputs the commands for the actuators. Even that all those works where successful with those monolithic implementations, their results were only applied to simple wheeled robots, being difficult to see how this type of neural controllers could control more complex robots with several sensors and actuators.

Because all those modularisation strategies focus on behaviour division, we say that they implement modularity at the strategic level.

3. Strategic and tactical modularity

From game theory [11], we think of strategy as the overall group of behaviours (or sub-goals) that a robot requires for the accomplishment of a goal, and of tactics as the actual means used to gain each of those sub-goals. Then, we define *strategic modularity* as the modular approach that identifies which sub-behaviours are required for an animat in order to obtain the global behaviour. This modularity can be performed from a distal or a proximal point of view, but it is a division that identifies a list of sub-behaviours. In contrast, we define *tactical modularity* as the one that creates the sub-modules required to implement a given sub-behaviour. Tactical modularity has to be implemented for each of the sub-behaviours by using the actual devices that will make the animat act, these are, its sensors and actuators. In tactical modularity, subdivision is performed at the level of the elements that actually are involved in the accomplishment of the sub-task.

To our extent, all of the works based on divide and conquer principles, focus their division at the strategic modularity level, it is, on how to divide the global behaviour into its sub-behaviours (or how to divide the task at hands into the required sub-tasks). Then, they implement each of those sub-behaviours by means of a monolithic neural controller. However, in this paper we propose the additional use of tactical modularity, where an additional decomposition is made at the level of the devices of the animat.

In some cases, it is possible to implement the whole behaviour required for the animat by only using tactical modularity, the use of one type of modularity does not imply the use of the other. In fact, we propose the use of both modularity types on a same controller as the required solution for complex behaviours in complex robots. Strategic modularity would decide which sub-behaviours are required for the complex global task, and tactical modularity would implement each of them on the complex robot.

We will not discuss here how to implement strategic modularity for a given robot and task, since it is not our goal. In principle, any of the modularisation methods used on the works described above is valid for its integration with our method of implementing tactical modularity.

3.1. Implementing tactical modularity

Tactical modularity should create modularity at the level of the robot devices which have to implement a required behaviour. It means that, once a sub-behaviour required for the animat has been decided, tactical modularity has to implement it using the sensors and actuators at hands. In this paper, we implement tactical modularity by creating a completely distributed controller composed of small processing modules around each of the robot sensor and actuator. We call this module an intelligent hardware unit (IHU) and its schematics is shown in figure 1a.

Every IHU is composed of a sensor or an actuator and an artificial neural network (ANN) that processes the information of its associated device (received sensor information for sensors, commands sent to the actuator for actuators). This means that the neural net is the one that decides which commands must be sent to the actuator, or how a value received from a sensor must be interpreted. All IHUs are interconnected to each other in order to be aware of what the other IHUs are doing. So, the net is also in charge of deciding what to say to the other elements as well as to interpret what the others are saying. The structure of a IHU can be seen in figure 1a, and figure 1b shows its application to a simple robotic system with two sensors and two actuators.

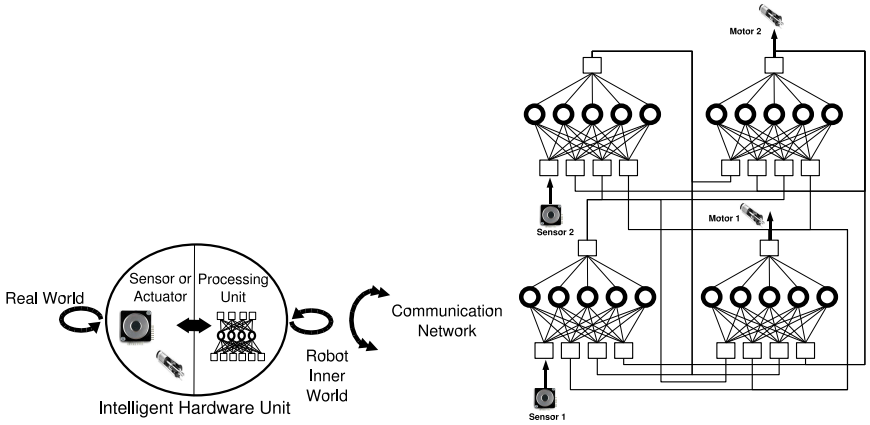

Figure 1. (a) IHU schematics (left) and (b) application of four IHUs to the control of a simple robot composed of two sensors and two motors (right)

Through the use of a neuro-evolutionary algorithm, IHU modules learn how to co-operate and coordinate between them, and how to control its associated element, allowing the whole robot to perform the sub-behaviour required. The algorithm selected is the ESP (Enforced Sub-Populations) [6][7], which has been proved to produce good results on distributed controllers [19].By using such algorithm it is possible to teach to the networks how they must cooperate to achieve a common goal (i.e. the sub-behaviour to implement), when every network has its own an different vision of the whole system.

A chromosome is generated for each IHUs' network coding in a direct way the weights of the network connections, and the whole group of neural nets is evolved at the same time with direct interaction with the environment.

4. Experimental framework

In order to test the theoretical approach presented in the previous section, we use a Khepera robot simulation as test-bed for our experiments. Experiments consists of the implementation of a tactical modular control system for the Khepera robot while performing a cleaning task, and the comparison of performance with other controllers developed previously by other researchers. The selected test-bed task is called the garbage collector as in [12]. In this task, a khepera robot is placed inside an arena surrounded by walls, where the robot should look for any of the cylinders randomly distributed on the space, grasp it, and take it out of the arena. Eventhough the robot used is not very complex, the task selected for it is complex. The garbage collector behaviour requires that the robot completely changes its behaviour based on a single sensor value change. When the robot does not have a stick on the gripper, its behaviour has to avoid walls, look for sticks, and approach them in order to pick them up. When the robot carries a stick, its behaviour has to change to the opposite, avoiding other sticks and approaching walls in order to release the stick out of the arena.

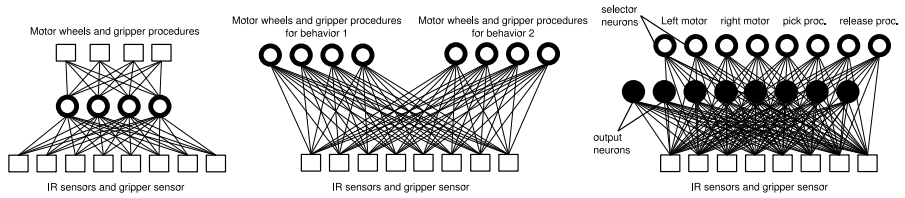

Figure 2. The first three architectures implemented: (a) a standard feedforward architecture, (b) a modular architecture with two sub-modules, and (c) an emergent modular architecture

4.1. The khepera robot and its environment

All the experiments reported for the khepera robot are done on a simulator. As simulator, we selected the freely available simulator YAST for the Khepera robot. It includes the simulation of the Khepera gripper, which is the turret capable of grasping objects. The Khepera gripper is composed of an arm that can be moved through any angle from vertical to horizontal, and two gripper fingers that can assume an open or closed position. The gripper is also composed of a sensor that indicates the presence of an object between the fingers.

Only the six front infrared sensors of the robot were used, as well as the gripper sensor. As actuators, the robot has two motors (left and right), but it is also possible to control the position of the gripper arm and the status of the gripper fingers (open or close). The control of the gripper is done by means of two procedures: the first procedure, when activated, moves the arm down, closes the gripper fingers and moves the arm up again, in order to pick a stick; the second procedure moves the arm down, opens the gripper fingers, and moves the arm up again, in order to release a stick.

The simulation setup is composed of a rectangular arena of 60x35 cm, surrounded by walls, and contains five garbage cylindric sticks. Each stick has a diameter of 2.3 cm and is randomly positioned inside the arena at every new epoch as well as the robot.

Experiments consist of 15 epochs of 200 time steps each, where an evolved controller is tested over the task. The duration of each time step is of 100 ms. Each epoch ends after the 200 steps or after a stick has been released out of the arena.

4.2. Neural architectures

Five different architectures were tested with the described setup. The first three architectures implemented are shown in figure 2. The first one, labelled (a), is a simple feedforward network with a hidden layer of four units. This architecture has seven inputs, corresponding to the six infrared sensors and the gripper sensor, and four outputs that correspond to the two wheel motors, and to the two procedures of control of the gripper (pick-up stick and release stick). The second architecture, labelled (b), is a modular architecture composed of two control modules. The modules were designed by hand from a distal point of view. One module controls the behaviour of the robot when it is looking for a stick, and the second module controls the robot when it carries one. The third architecture (c) is the emergent modular architecture as defined by Nolfi [12]. In this case, there are two different modules with seven inputs (the seven sensor values) and four pairs of output neurons. The four pairs outputs of the first module code for the speed

of the motors (left and right) and the triggering of the pick-up and release procedure. The outputs of the second module determine which of the two competing neurons of the first module had actual control over its corresponding effector. Those three architectures were used by Nolfi on his work and are implemented here in order to verify that the setup implemented produces similar results to that obtained by Nolfi, and by hence, allow us to compare results.

The two other architectures are the ones that use tactical modularity. The first one, which we will call architecture (d), is a direct application of tactical modularity over the garbage collector task. In this case, only one global behaviour is required for the task (it is, the garbage collector behaviour), and we implement that behaviour using tactical modularity, it is, the creation of one IHU element for each device involved. Since eleven devices are involved (seven sensors and four actuators), we use eleven IHUs for the construction of the controller. No figure is provided for this architecture, because of its complexity, but it would look like figure 2b with eleven devices. We create a IHU for each of the infra-red sensors and the gripper sensor, and four IHUs for the left and right motors, and the two gripper procedures. Each IHU is implemented by a feedforward neural net with eleven inputs, no hidden units, and one output.

The second architecture we will call architecture (e). In this case, we apply the two modular approaches to solve the control problem. On a first stage, we use strategic modularity to define the required sub-behaviours for the global task. We manually decide to divide the global behaviour of the animat into two sub-behaviours: the first sub-behaviour should look for a stick while avoiding walls, and once detected, pick it up. The second sub-behaviour should be activated once a stick is picked up, and it has to avoid other sticks and go to the wall where leave out the carrying stick. On the second stage, each of those strategic modules is implemented using tactical modularity, of the same type as in the previous architecture (d), but now focused on those more limited sub-tasks. This means that we will obtain two different strategic modules, each one implementing one sub-behaviour by means of 11 tactical modules. Training of each strategic module is performed separately, and once they are evolved, they are joined for the final global controller.

5. Results

We evolved all the architectures using our evolutionary setup. The fitness function rewarded those controllers that were able to release one stick out of the arena. Controllers that were able to pick up one stick were also rewarded with a lower fitness. We included an additional term in the fitness function that decreased the fitness when the stick was released inside the arena. Like Nolfi did, we implemented a mechanism that artificially added a stick in front of the robot each time it picked up one stick, in order to increase the situations where the robot encounters an obstacle in front of it while carrying a stick. The evolutionary process was performed ten times for each architecture, and the results presented here show the averaged fitness obtained from those ten runs.

The results, plotted in figure 3a, show that architectures (a), (c) and (d) were able to reach the maximal performance of releasing a stick in every epoch, and being the only difference the number of generations that each architecture required to reach that maximal performance. However, architecture (b) was not able to reach maximal performance.

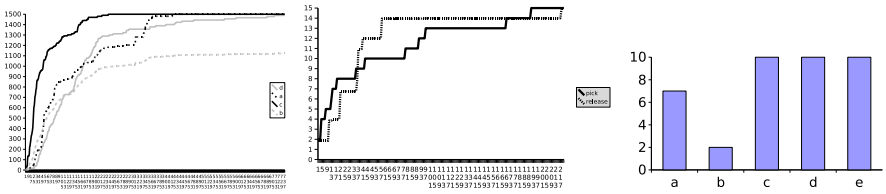

Figure 3. (a) Evolution of the fitness obtained by the best controller of each architecture (left). (b) Evolution of the fitness obtained by the best controller of each strategic module of the architecture (e) (middle). (c) Number of times that the best controller of each architecture was able to correctly recollect the five sticks without displaying any error (right).

These results are consistent with the ones reported by Nolfi, and show that architectures not based on distal modularization obtain better results.

Evolution of architecture (d) was performed in the same conditions as in (a), (b) and (c), and as can be seen in figure 3a, took a longer number of steps to evolve the same behaviour and about the same performance as in (a) and (c).

On the other side, we evolved architecture (e) in two stages. On the first stage, we evolved the strategic module in charge of avoiding walls and picking a stick. On a second stage, we evolved the strategic module in charge of avoiding sticks and approaching walls to release the stick. Both strategic modules were evolved using tactical modularity. Figure 3b shows the number of evolutions required for the controllers to generate those behaviours in both cases. Once the two strategic modules were evolved, they were used on a global controller which used both modules, switching between one and the other to govern the robot depending on the status of the gripper sensor. The final behaviour obtained correctly performed the garbage collector behaviour.

6. Discussion

The results obtained in the previous section show that, except in architecture (b), all the architectures are capable of a garbage collector behaviour with a 100% success rate. Only the evolutionary time required for each architecture shows a difference between them, since their behaviours are very similar, including errors about thinking that a stick is a wall that sometimes appear in the final controllers. We conclude then, that the strategic and tactical approaches are as good as the others on the garbage collector task.

Having shown that the all the architectures are able to perform the garbage collector behaviour,the next question may be what is the point about using strategic and tactical modularity on a robot controller. The reasons are two: first, the use of tactical modularization allows the introduction of new sensors and actuators at any time. This point has been proved on a previous work [18], where aditional sensors and actuators of an Aibo robot where introduced at different steps of the evolutionary process. To our extent, no other architecture is able to do that.

Second, the results obtained with tactical modularization are more robust. In order to probe that,we perform an additional test. We visually compare the performance obtained in all the architectures when performing the garbage collector. For this, the best neural nets obtained for each architecture on each run, were tested into the arena, to see how

many times were able the controllers to pick up and release outside the arena the 5 sticks within 5000 cycles without displaying any incorrect behaviour (those are, crashing into walls, trying to grasp a wall or trying to release a stick over another). The results are shown on figure 3c.

7. Conclusions

We have proposed two types of modularisation required for the generation of complex behaviours in complex robots. We have shown the differences between them and how they can be combined for the generation of complex behaviours in robots.

Modularisation at the level of behaviour has already been widely used by other researchers. The real new thing provided by this research is the use of a new level of modularisation focused on the robot devices (tactical modularity), and the demonstration that the use of both types of modularisation together leads to more complex behaviours in more complex robots.

Future work includes the application of strategic and tactical modularity for complex behaviours in complex robots. This work has already successfully started with the application of tactical modularity to an Aibo robot for the generation of a standing up behaviour [17] and a walking behaviour [18].

References

[1] G. Auda and M. Kamel. Cmnn: Cooperative modular neural networks for pattern recognition. In *Pattern Recognition in Practice V Conference*, 1997.

[2] F. Azam. *Biologically inspired modular neural networks*. PhD thesis, Virginia Polytechnic Institute and State University, 2000.

[3] K. Chen and H. Chi. A modular neural network architecture for pattern classification based on different feature sets. *International Journal of Neural Systems*, 9(6):563–581, 1999.

[4] Ian Davis. *A Modular Neural Network Approach to Autonomous Navigation*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, April 1996.

[5] R. Di Ferdinando, A. Calabretta and D. Parisi. Evolving modular architectures for neural networks. In *Proceedings of the sixth Neural Computation and Psychology Workshop: Evolution, Learning and Development*, 2000.

[6] F. Gomez and R. Miikkulainen. Incremental evolution of complex general behavior. Technical Report AI96-248, University of Texas, 1, 1996.

[7] F. Gómez and R. Miikkulainen. Solving non-markovian control tasks with neuroevolution. In *Proceedings of the IJCAI99*, 1999.

[8] R. Jacobs, M. Jordan, S.J. Nowlan, and G. E. Hinton. Adaptative mixture of local experts. *Neural Computation*, 1(3):79–87, 1991.

[9] B. Lara, M. Hülse, and F. Pasemann. Evolving neuro-modules and their interfaces to control autonomous robots. In *Proceedings of the 5th World Multi-conference on Systems, Cyberbetics and Informatics*, 2001.

[10] P. Manoonpong, F. Pasemann, and J. Fischer. Modular neural control for a reactive behavior of walking machines. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2005.

[11] Roger McCain. *Game Theory : A Non-Technical Introduction to the Analysis of Strategy*. South-Western College Pub, 2003.

[12] S. Nolfi. Using emergent modularity to develop control systems for mobile robots. *Adaptative Behavior*, 5(3-4):343:364, 1997.

[13] S. Nolfi. Evolutionary robotics: Looking forward. *Connection Science*, 4:223–225, 2004.

[14] R W. Paine and J. Tani. How hierarchical control self-organizes in artificial adaptive systems. *Adaptive Behavior*, 13(3):211–225, 2005.

[15] D. Parisi R. Calabretta, S. Nolfi and G. P. Wagner. Emergence of functional modularity in robots. In J.-A. Meyer R. Pfeifer, B. Blumberg and S.W. Wilson, editors, *Proceedings of From Animals to Animats 5*, pages 497–504. MIT Press, 1998.

[16] J. Tani and S. Nolfi. Learning to perceive the world as articulated: an approach for hierarchical learning in sensory-motor systems. *Neural Networks*, 12:1131–1141, 1999.

[17] R. Tellez, C. Angulo, and D. Pardo. Highly modular architecture for the general control of autonomous robots. In *Proceedings of the 8th International Work-Conference on Artificial Neural Networks*, 2005.

[18] R. Téllez, C. Angulo, and D. Pardo. Evolving the walking behaviour of a 12 dof quadruped using a distributed neural architecture. In *Proceedings of the 2nd International Workshop on Biologically Inspired Approaches to Advanced Information Technology*, 2006.

[19] H. Yong and R. Miikkulainen. Cooperative coevolution of multiagent systems. Technical Report AI01-287, Department of computer sciences, University of Texas, 2001.

[20] J. Ziemke, T. Carlsson and M. Bodén. An experimental comparison of weight evolution in neural control architectures for a 'garbage-collecting' khepera robot. In F. Löffler, A. Mondada and U. Rückert, editors, *Experiments with the Mini-Robot Khepera - Proceedings of the 1st International Khepera Workshop*, pages 31–40, 1999.