In IEEE International Conference on Fuzzy Systems (Fuzzy2000), San Antonia, Texas, USA, 7-10 May, 2000

Online Learning of Fuzzy Behaviour Co-ordination for Autonomous Agents using Genetic Algorithms & Real-Time Interaction with the Environment

Hani Hagras, Victor Callaghan, Martin Colley.

The Computer Science Department, Essex University
Wivenhoe park, Colchester, CO43SQ, England.
(Email: hakhag@essex.ac.uk)

Abstract- **This paper addresses the development of a system for online learning of fuzzy behaviour co-ordination for autonomous agents in the form of robots based on genetic algorithms (GAs) and real-time interaction with the environment. The proposed system organises the behaviours hierarchically and uses fuzzy engines to implement both the behaviours and their co-ordination mechanism. In previous work we reported on our success in the online learning of individual behaviours (rules and membership functions)[4]. In this paper we report on a system that allows the fuzzy Membership Function (MF) for behaviour co-ordination to be learnt online in a manner that satisfies some high level mission or plan. The GAs uses adaptive learning parameters and guided constrained optimisation to speed the GAs search and enable it to be performed via real-world interaction rather than off-line simulation. The results of this work are compared with results reported elsewhere and reveals this approach to have a superior learning performance while learning using real outdoor robots in changing environments. The ability to learn co-ordination skills in a short time interval without human intervention makes this approach particularly useful for applications where access is difficult such as nuclear reactors, underwater vehicles and space robots and fast changing and dynamic environments such as the agricultural environments.**

I. INTRODUCTION

Since the first Brooks's seminal papers [2], many autonomous agents have been implemented following the behaviour-based paradigm, where the overall operations of an agent arises from the interaction of basic independent behaviours. The importance of the problem of behaviour co-ordination was noticed; i.e. how to co-ordinate the simultaneous activity of several independent behaviour-producing units to obtain an overall coherent behaviour that achieves the intended task. Early behaviour-based architectures [2] relied on a fixed arbitration policy, hard wired into a network of suppression and inhibition links. This rigid organisation contrasts with the requirement that an autonomous robot can be programmed to perform a variety of different tasks in a variety of different environments [10]. Later proposals relied on dynamic arbitration policies, where the decision of which behaviour(s) to activate depends on the current goals given by the planner and environmental contingencies. However, many of these ideas do not allow for the concurrent execution of behaviours [10]. Both fixed and dynamic arbitration policies can be implemented using the mechanisms of fuzzy logic. The two main advantages in doing so are the ability to express partial and concurrent activation of behaviours and the smooth transitions between behaviours.

One of the main problems for the behaviour-based approach to agent design concerns the identification of the best combination of the most suitable basic behaviours to achieve a task, in a given situation. This is because it is difficult to predict the interaction between the different behaviours or their best combination especially when large number of behaviours are involved. Some approaches utilising machine-learning have been proposed. Mahadevan [6] proposes a system that learns the basic behaviours most suitable for a given, predefined behaviour architecture. Dorigo [3] learns to co-ordinate behaviours organised in different hierarchical architectures. Bonarini [1] learns a coordinator, implemented by fuzzy rules. All of these systems have learnt the relationship among antecedent and consequent values, but keep constant the values of the membership. In other terms, it learns the structure of the behaviour, but not the interpretation of the data it uses. With regard to our work it is important to note that all these approaches were implemented using simulation, even if tested on real robot, the learning was performed offline in simulation and then the "learnt controller" was downloaded to the real robot.

Broadly speaking, our work situates itself in the recent line of research which concentrates on the realisation of artificial agents strongly coupled with the physical world which we refer to as embedded-agents. A first fundamental requirement is that such agents must be grounded in that they must be able to carry on their activity in the real world. Another important point is that adaptive behaviour cannot be considered as a product of an agent considered in isolation from the world, but can only emerge from strong coupling of the agent and its environment [3]. There are several reasons why those who want to use simulation models to develop control systems for real robots may encounter problems [9]. For example simulations do not usually consider all the physical laws governing the interaction of a real agent with its environment, such as mass, weight, friction, etc. Also physical sensors deliver uncertain values, and the commands to the actuators have uncertain effects, whereas simulative models often use grid-worlds and sensors which return perfect information. Also different physical sensors and actuators, even if apparently identical, may perform differently because of slight differences in electronics and mechanics or because of their different positions on the robot.

Fuzzy logic has become a popular approach to reactive robot control in the recent years. Given the uncertain and incomplete information an autonomous robot has about the environment, fuzzy rules provide an attractive means for mapping sensor data to appropriate control actions in real time. The success of fuzzy control is owed in large part to the technology's ability to convert qualitative linguistic descriptions into non-linear mathematical functions. In addition fuzzy controllers exhibit robustness with regard to noise and variations of system parameters.

The aim of our research is to produce intelligent machines where self-adaptation is important. For instance for applications where programming costs are a factor (e.g. niche applications, dynamic environments, or multi-purpose roles such as agriculture etc) or inaccessible environments (e.g. underwater or outer space). In these environments it is required to perform online learning through interaction with the real environment in a short time interval. Such an approach both saves money and increases reliability by allowing the robot to automatically adapt to the changing user and environment needs throughout its lifetime without further human programming. In our previous work [4,5] we reported on our success in the online learning of individual behaviours (rules and membership functions) using a Fuzzy-Genetic embedded-agent. In this paper we are looking for the best combination of these behaviours in order to achieve the goals supplied by a high level plan. The reported behaviour co-ordination is based on the use of a modified form of GAs. GAs have been successfully applied to solve a variety of difficult theoretical and practical problems by imitating the underlying processes of evolution such as selection, recombination and mutation. Using only an objective function, GAs can adapt a system to deal with any desired task without the need for a mathematical model. These reasons make GAs suitable for learning in robotics as it is considered essentially difficult if not impossible to formulate a mathematical function to adequately describe the robot, the physical world and their interactions. GAs are criticised for being slow because they require significant population of robots for fitness testing, and the robots must be tested over many generations. Much of the learning using GAs reported in the literature was conducted via offline simulation as the normal evolutionary time scale needed makes online use infeasible. In this paper we introduce techniques for allowing GAs based mechanism to be applied in online systems without the need for simulation. The techniques involve the use of adaptive learning parameters, guided GAs optimisation plus the adoption of nearness criteria for allowing acceptance of reasonable solutions.

In the next section we will discuss the fuzzy hierarchical architecture and then we introduce the GAs behaviour co-ordination learning,. We finish by presenting experimental results, conclusions and future work.

II. Fuzzy Hierarchical Systems

Most commercial fuzzy control implementations feature a single layer of inferencing between two or three inputs and one or two outputs. For autonomous vehicles, however the number of inputs and outputs is usually large and the desired control behaviours are much more complex. The mapping can be made manageable by breaking down the input space for analysis by multiple agents, each of which responds to specific types of situations and then integrating the recommendations of these agents. Agents, also called behaviours, can be designed independently to exhibit behaviours such as goal seeking, obstacle avoidance, and edge following [10]. The vehicles we have being using in our indoor experiments have eight inputs (7 sonar inputs and an infrared bearing sensor) and two outputs which are left and right wheel speeds (steering and speed in case of outdoor vehicles). And 8-axis vectored bump switch, the hardware is based on embedded Motorola processors (68040) running VxWorks RTOS. Each control cycle takes about 100 ms. In our previous work we have shown how our hierarchical fuzzy engine approach can reduce the number of rules by two orders of magnitude [4]. This hierarchical arrangement is shown in Figure (1) co-ordinating 4 behaviours namely Left edge following, right edge following, obstacle avoidance and goal seeking. Our work uses a method similar to the methods suggested by Saffiotti [10] and Tunstel [12] to perform the high level co-ordination between such behaviours based on using fuzzy logic to both implement individual behaviour elements and the necessary arbitration. Each single behaviour is viewed as an independent fuzzy controller. In this hierarchical architecture we will use a fuzzy operator to combine the preferences of different behaviours into a collective preference. According to this view, command fusion is decomposed into two steps: preference combination and decision. In the case of using fuzzy numbers for preferences, product-sum combination and height defuzzification, the final output equation is given by [10]:

$$\mathbf{C}= \frac{\sum_i (BW_i * C_i)}{\sum_i BW_i} \qquad (1)$$

Here i = right behaviour, left behaviour, goal seeking, C_i is the behaviour command output. BW_i is the behaviour weight. The behaviour weights are calculated dynamically taking into account the context of the mobile robot. In Figure (1) each behaviour is treated as an independent fuzzy controller and then, using fuzzy behaviour combination, we obtain a collective fuzzy output which is then deffuzzified to obtain a final crisp output. There are some few parameters that must be calculated in the Root Fuzzy System block in Figure (1). These parameters are the minimum distance of the front sensors which is represented by d1, the minimum distance of the left side sensors which is represented by d2, the minimum distance of the right side sensors represented by d3. And the minimum of the fuzzy MF of d1,d2,d3 represented by d4, which reflects how the robot has no obstacles around. After calculating these values, each of them is matched to its MF which is shown in Figure(2). The values of A, B are the base values to be learnt by the GAs for d1 and C, D in case of d2 and E and F in case of d3 and G and H in case of d4. After matching d1,d2,d3,d4 to their MF, we have fuzzy values. These fuzzy values are used as inputs to the context rules which are suggested by the high level planner according to the mission to be performed by the robot. Which are in general : *IF d1 IS LOW THEN OBSTACLE AVOIDANCE. IF d2 IS LOW THEN LEFT EDGE FOLLOWING. IF d3 IS LOW THEN RIGHT EDGE FOLLOWING. IF d4 IS HIGH*

THEN GOAL SEEKING. The context rules determines which behaviour is fired, and to what degree, depending on the fuzzy MF in Figure (2). Then the final robot output is a mixture of the different behaviour outputs each weighted by the degree of its importance calculated using Equation (1).

Note that a planner can eliminate the unneeded behaviours from the context rules according to the robot's mission. For example in case of corridor following, there is no need for goal seeking so the last rule in the context rule is deleted and the goal seeking block is removed from Figure (1). We always use the obstacle avoidance behaviour as a safety feature. Note that the robot mission affects only the context rules by deleting some of its rules not adding to them, and the robot mission is represented by the GAs objective function.

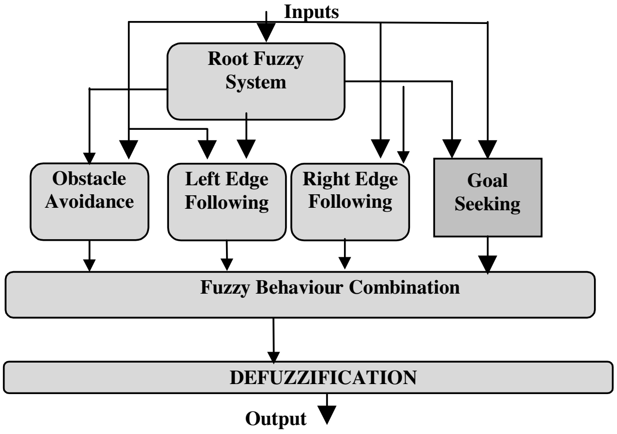

Figure 1. The behaviour co-ordinated system.

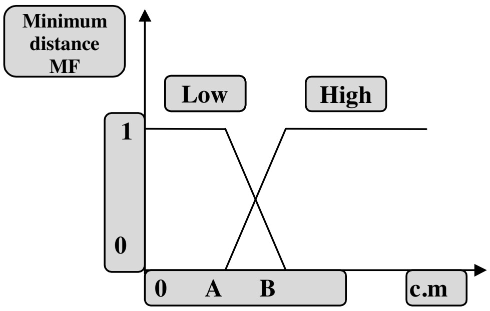

Figure 2. The MF for the minimum front and side sensors.

III. THE GENETIC ALGORITHMS LEARNING

A. Population Initialisation

Each parameter to be identified in the MF which are A, B in case of d1 and C,D, in case of d2 and E,F in case of d3 and G,H in case of d4 is represented by 5 bits (which were found empirically to be the minimum number of bits to achieve reasonable results and low computation time). Note that if a behaviour is deleted by the planner then its parameters will not be encoded. These parameters are aligned together to form a chromosome which represents a possible solution for the problem. This means that when all the behaviours are present we have at maximum a 4*2*5 =40 bit chromosome. We use the concept of GAs guided constrained optimisation [11] which incorporates human heuristic knowledge into the optimisation algorithm. All the chromosomes are initialised within the sensor range, which is fair assumption as it is very easy to know the maximum and minimum sensors ranges from the manufacturer data sheet. In any case, the algorithm is also capable of determining the sensor ranges, but it will take longer time to converge to a solution (as will be shown later). Also we use ordering of the MF so that during optimisation the GAs is forced to produce ordered parameters (for example A is always greater than B and C is greater than D, etc) within the sensor range.

B. Fitness Evaluation

After initialisation of the chromosomes the robot starts moving to test the proposed solution. The determination of the fitness function depends on the high level mission. The high level missions can be regarded as a deviation minimisation problem. For example in case of corridor following mission, aligning to the centre line can be obtained by minimising the deviation between the left front and the right front sensors to reach a *nominal value* of zero and trying to achieve this by minimum steering actions and maximum speed. Also in case of following an edge while avoiding obstacles and reaching a goal at the end of this edge, the mission can be viewed as minimising the deviation from a *nominal* desired distance in following the edge "and" (and function is represented by mathematical sum) minimising the deviation from a *nominal* safe distance in avoiding obstacles and minimising the deviation from the goal to a *nominal value* of zero with minimum steering deviation and a high speed. The deviation is measured by the robot physical sensors (sonar in case of edge following and infrared scanner in case of goal seeking). This gives the robot the opportunity to adapt to the sensor imprecision.

The chromosome fitness is evaluated by how much it reduces the average of the total absolute deviation, di (which is calculated as the sum of the individual deviations over the existing behaviours *m, in a given mission* at each control step) while using minimum steering deviation and high speed, where k is a given behaviour and d_i is given by:

$$d_i = \sum_{k=1}^{m} \left(\frac{| nominal value - deviated.value |}{\max .deviation} \right)_k \qquad (2)$$

Where the nominal value will correspond to the value that reflects the mission goals; the deviated value is any value deviating from the nominal value. The maximum deviation corresponds to the maximum deviation that can occur. The solution Fitness is given by:

$$\text{Fitness} = \text{constant} + \left(1 - \frac{\sum_{i=1}^{N} d_i}{N}\right).(1-F).V \qquad (3)$$

Where N is the total number of control steps done till the end of testing the solution. F is the average normalised (with respect to the maximum steering) steering of the robot over N Control steps. V is the average speed of the robot over N steps. This maximises the fitness by minimising the total absolute deviation and the steering deviation and maximising the speed.

C. Generation of New solutions

We use here a population formed of 4 chromosomes which was found to be the smallest population to give convergence whilst maximising computational simplicity and real time performance. When all the solutions have determined a fitness value, they are ready to generate new solutions.

We used Mandavilli method [7] to adapt the control parameters (mutation and crossover probabilities). The strategy used for adapting the control parameters depends on the definition of the performance of the GAs. In a non-stationary environment (which is the case for outdoor and dynamic environment), where the optimal solution changes with time, the GAs should possess the capacity to track optimal solutions. The adaptation strategy needs to vary the control parameters appropriately whenever the GAs is not able to track the optimum. It is essential to have two characteristics in GAs for optimisation. The first characteristic is the capacity to converge to an optimum (local or global) after locating the region containing the optimum. The second characteristic is the capacity to explore new regions of the solution space in search of the global optimum. In order to vary P_c (crossover probability) and P_m (mutation probability) adaptively, for preventing premature convergence of the GAs, it is essential to be able to identify whether the GAs is converging to an optimum. One possible way of detecting convergence is to observe the average fitness value f' of the population in relation to the maximum fitness value f_{max} of the population. f_{max}-f' is likely to be less for a population that has converged to an optimum solution than that for a population scattered in the solution space. The equations that determines P_c, P_m are given by :

$$P_c = (f_{max}-f'')/(f_{max}-f') \quad f''\geq f' \; , \; P_m = (f_{max}-f)/2.(f_{max}-f') \quad f\geq f' \qquad (4)$$
$$P_c = 1 \quad f''<f' \; , \; P_m = 0.5 \quad f''<f' \qquad (5)$$

Where f'' is the larger of the fitness values of the solutions to be crossed, f is the fitness of the individual solutions. The method means that we have P_c and P_m for each chromosome. The type of crossover was chosen to be a single point crossover for computational simplicity and real time performance. In [7] this method was superior to the simple GAs and gave a faster convergence rate of 8:1. We use this adaptive method for finding the values of crossover and mutation probabilities. This approach leads to fast convergence, and enables adaptation in changing environments relieving the designer from the usual need to determine these values, heuristically [8,14].

We use an elite strategy, meaning that the best individual is automatically promoted to the next generation, and used to generate new populations. Also the GAs is constrained to give ordered solutions within the sensor range so that we can minimise the search space of the GAs and achieve faster conversion.

In order to justify these techniques we have conducted various experiments with this Adaptive Genetic Algorithm (AGA) with open range, and AGA with constrained range and Simple genetic algorithm (SGA) with constrained range for the problem of corridor following (using an indoor robot). The SGA was tried with differing parameters in the range [0.5 1.0] for P_c and [0.001 0.1] for P_m. It was found that the AGA converges to a solution in average after *only 7 iterations and 8 minutes of the real indoor robot time* (most of the time is consumed by moving forward to test the solution and then moving backward to the same position). The AGA with open range converges after a larger number of iterations (11 iterations in average) as it needs more time to explore the search space and determine its limits. This takes about 18 minutes of our robot's time to converge to a solution. The SGA (using the optimal found parameters P_c=0.68 and P_m=0.1) with defined limits converges to a solution after an average of 32 iterations and 40 minutes of our robot's time. Thus, we conclude from these experiments that the use of our constrained AGA mechanism makes online learning practical.

D. Ending Criteria

The robot finishes learning a given mission when there is a solution satisfying the criteria that all the average absolute deviations from the nominal values for each behaviour k in a given mission fall within the tolerance of sensors used in measuring this deviation. For example in case of reaching a goal via following an edge while avoiding obstacles, the ending criteria will be when the average absolute deviation from the desired distance for the edge following behaviour reaches 10% (the degree of imprecision associated with the sonar sensors used); and the average absolute deviation from the safe distance for the obstacle avoidance behaviour reaches 10% (the degree of imprecision associated with the sonar sensors used); and when the average absolute deviation from the goal for the goal seeking behaviour reaches 13% (the degree of imprecision associated with the infrared scanner and beacons sensors used).

IV. EXPERIMENTAL RESULTS

All the following experiments were performed using real robots and the robot response was drawn by a bottle of paint fixed at the back of the robot in case of indoor robots and a tape connected to the left back wheel in case of the outdoor robots. We had performed some experiments using the indoor robots and the same algorithm was then transferred to the outdoor robots operating in an open outdoor changing environment which demonstrates the portability of our algorithm and that the algorithm parameters are robot independent.

Figure (3-a) shows the robot trying to learn the best way of coordinating three behaviours which are left edge following, right edge following and obstacle avoidance in order in perform the high level mission of aligning to the centre line of a corridor. Note that this mission could be performed by coordinating only left and right edge following behaviours but we have added obstacle avoidance as a safety feature.

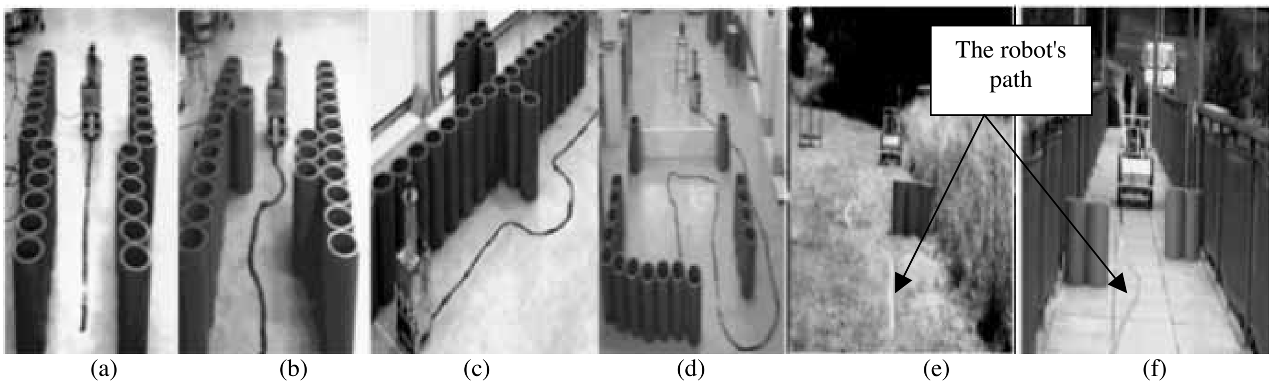

Figure 3: a) learning aligning to a corridor. b)following a complex corridor. c) learning right edge following while avoiding obstacles. d) learning escaping a maze, avoiding obstacles and seeking a goal. e) outdoor robot learning to follow an irregular corridor whilst avoiding obstacles. f) avoiding obstacles and aligning to the centre of the corridor.

The robot had found a solution after an average of 7 iterations over 5 runs starting from different positions (5 runs were chosen as the average over a higher number of runs tends to be more or less the same as 5) which took about 8 minutes of robot time. The robot had followed the centre line of the corridor with a small average deviation of 1.05 cm and a standard deviation of 0.8. The learning system restricted the obstacle avoidance behaviour to a near range of 18.75 to 25 c.m, and the left and right edge following were given high range to control the vehicle in a wide range of corridors within the sensors range (1m). The values of the left and right edge following co-ordination parameters are different because of the differing sonar characteristics and the different robot kinematics, but the values are close because the behaviours are similar but in opposite sides. In Figure (3-b) we tried the robot in a complicated corridor (this experiment was suggested by Bonarini [1]), in which the corridor includes obstacles, the robot again followed the centre line of the corridor with a small average and standard deviation.

In Figure (3-c) the robot learnt to co-ordinate the right edge following and the obstacle avoidance behaviours in order to perform the mission of right edge following whilst keeping a specified distance of 20 c.m and avoiding obstacles and keeping a safe distance of 25 c.m. The robot converged to a solution after an average of 6 iterations taking 7 minutes of robot time. It achieved a solution involving a small average and standard deviation for the both the desired edge following and the safe obstacle avoidance distances. The learning system had enlarged the range of obstacle avoidance behaviour in order to deal as soon as possible with any obstacles and avoid them at a safe distance. The range of the left edge following remained still large but different from the previous experiments as the mission was different.

In Figure (3-d) the robot is given a mission to escape a maze (by giving penalty for 360^o motion) and reach a goal whilst keeping a safe distance from obstacles. This is a local minimum problem set by Voudouris [13]. We solved the problem simply by co-ordinating left (or right edge following) and obstacle avoidance and goal seeking behaviours. We allowed our system to learn the best suited behaviours for this mission and thus we activated all the behaviours (left/right edge following and obstacle avoidance and goal seeking). The robot converged to a solution after an average of 6 iterations taking 7 minutes of the robot time. The robot chose to activate left edge over a large range while it had truncated right edge following. The robot selected left edge following as providing a shorter path to the goal (in other geometrical settings it may have chosen right edge following). This experiment shows that the system is able to choose the most appropriate behaviours for a given mission.

In Figure (3-e) we used an outdoor robot with 7 sonar sensors, its hardware was also based on embedded Motorola processors (68040) running VxWorks RTOS with a motor for motion and another for steering. We aimed to test the above indoor environment techniques in more dynamic and challenging outdoor environment involving systems with different mechanical and sensor characteristics.

In Figure (3-e) we applied the same mission that was applied to the indoor robot in Figure (3-c) which is following an irregular crop at a desired distance of 120 c.m whilst avoiding obstacles at a distance of 1m. The robot converged to a solution after an average of 6 iterations taking 3 minutes of robot time (this robot is faster than the indoor robot). It gave a small average deviation of 2.4 cm and a standard deviation of 1.3 for the edge following, an average deviation of 2.7 cm and standard deviation of 1.5 for the desired safe distance.

In Figure (3-f) we applied the robot to learning the same mission learnt by the indoor robot in Figure (3-a). This involved aligning the robot to the centre line of an outdoor irregular corridor whilst avoiding obstacles at a safe distance of 1m. The robot converged to a solution after an average of 8 iterations taking 4 minutes of the robot time with small average deviation and standard deviation from the desired values.

All of the above experiments were performed in changing weather conditions varying from rain through sunshine to wind and on different ground surfaces. The robot was always able to fulfil its objective and produce the same response under these different conditions. This was because the co-ordination parameters were learnt online and through interaction with the real environment so that the system could

In IEEE International Conference on Fuzzy Systems (Fuzzy2000), San Antonia, Texas, USA, 7-10 May, 2000

deal with different environments taking into account the sensor imperfections. The algorithm succeeded in learning the appropriate parameters in such challenging environments as an agricultural setting without human intervention, where it is difficult for a human designer to estimate the coordination parameters as he has to take into account, the irregularities of the crop and the ground and environmental conditions. This system can also be applied to environments which are difficult to access such as the nuclear reactors, space robots and under water vehicles, as it needs the only a desired objective and then it will adjust itself. In all the above experiments the system produced different solutions for the same mission, however the average and standard deviation of the worst solution (in terms of deviation) was close to those produced by the best solution, which supports our thesis that this techniques is capable of providing good behaviour coordination. We also performed the statistical t-Test for matched (paired) samples over the worst and best solution and we found that the t-Test had showed that the two solutions are statistically similar. Also, the hand crafted coordination parameters always produced much larger average and standard deviation (3 times greater) than the automatically learnt ones. In addition, the learnt solution gave an average path repeatability of 98 % when starting from different positions.

Although it is difficult to compare our work with other researchers, as each group uses different robots, sensors and different hardware but we have tried to compare the concepts and obtained results. Bonarini [1], used evolutionary learning to learn to co-ordinate fuzzy behaviours for autonomous agents. This system was implemented in simulation and did not use real robots, unlike our system. His system learnt rules in the co-ordinator and kept constant the MF, which learnt the structure of the behaviour but not the interpretation of the data it uses. In case of the corridor following in experiment (3-b) this caused the robot not to stay in the centre of the corridor whilst we were able to balance the behaviours to cause aligning to the centre line of the corridor. Also he did not consider applying his system to more complicated missions such as we have. We have faster conversion for our system while learning online.

Dorigo [3] used a classifier system to learn the co-ordination strategy for a robot, in which the co-ordination is performed by a bit string. He used a real robot to learn simple tasks like goal-seeking. It took him 4 hours to achieve convergence, while the maximum time for our system to learn more complicated missions was 9 minutes.

Mahadevan [6] used reinforcement and Q-learning for learning basic behaviours aimed at simple tasks such as finding a box and pushing it. After two hours of learning (compared with maximum learning time of 9 minutes for more complicated tasks for our system), the performance of the robot at the end learning according to Mahadevan is somewhat disappointing.

V. CONCLUSIONS

In this paper, we have presented a fast converging online-learning GAs based system that learns the best co-ordination parameters of fuzzy behaviours within a hierarchically organised architecture for a given mission or plan. The GAs had used adaptive learning parameters, guided constrained GAs optimisation [11] and a “stopping window” criteria. These techniques have enabled the GAs to converge in a short interval (a maximum of 4 minutes using our outdoor robots). The system learnt co-ordination of its behaviours to perform different complicated tasks and the system was always able to find the best co-ordination membership parameters to perform the mission. It was also proved able to identify the best suited behaviours for a given mission. The system is appropriate for inaccessible environments such as underwater or space environments as well as dynamic and changing environments such as the agricultural environments. The techniques are advantageous wherever programming costs are a factor during the agent’s lifetime as the system effectively programs itself. For the future work, we intend to integrate some form of planning with our system. We also intend to implement the patented aspects of the system (UK patent number “99 10539.7) in programmable hardware in order to meet the needs of very fast real-time needs or cost conscious applications.

REFERENCES

[1] A. Bonarini, F. Basso, “ Learning to coordinate fuzzy behaviours for autonomous agents”, International Journal of Approximate Reasoning, Special issue on "Genetic Fuzzy Systems for Control and Robotics", F. Herrera (Eds), 17 (4), pp. 409-432, 1997.
[2] R. Brooks , “Intelligence without representation”, Artificial Intelligence 47, pp. 139-159, 1991.
[3] M. Dorigo, “ Robot shaping : developing autonomous agents through learning”, Artificial Intelligence, V.71, pp..321-370, 1995.
[4] H.Hagras, V. Callagahan, M. Colley, “Online Learning of Fuzzy Behaviours using Genetic Algorithms & Real-Time Interaction with the Environment”, 8th IEEE International Conference on fuzzy systems, Seoul –Korea, pp. 668-673, 1999.
[5] H.Hagras, V. Callaghan, M.Colley and M. Carr-West, “ A Fuzzy-Genetic Based Embedded-Agent Approach to Learning and Control in Agricultural Vehicles”, 1999 IEEE International Conference on Robotics and Automation, Detroit-USA, pp. 1005-1010, 1999.
[6] S. Mahadevan, J.H.Connell, “ Automatic programming of behaviour-based robots using reinforcement learning”, Artificial Intelligence, V. 55, pp. 311-365, 1992.
[7] S.Mandavilli, M.Patnaik “ Adaptation in Genetic Algorithms”, In Genetic Algorithms for Pattern Recognition, S. Pal (Eds), 1997.
[8] V.Matellan, “Genetic learning for fuzzy reactive controllers”, Journal of Robotics and Autonomous systems”, V25, pp. 33-41, 1998.
[9] O. Miglino , H. Lund, S. Nolfi., “ Evolving Mobile Robots in Simulated and Real Environments. Artificial Life” Technical Report NSAL-95007, Reparto di Sistemi Neurali e Vita Artificiale, Istituto di Psicologia, Consiglio Nazionale delle Ricerche, Roma, 1990.
[10] Saffiotti, “ Fuzzy Logic in Autonomous Robotics : behaviour coordination.”; 6th IEEE International Conference on fuzzy systems, Barcelona-Spain, pp. 573-578, 1997.
[11] G. Tan, X. Hu, “ More on Designing Fuzzy Controllers using GAs: Guided Constrained Optimization”, 6th IEEE International Conference on fuzzy systems, Barcelona-Spain, pp. 497-502, 1997.
[12] E. Tunstel, T. Lippincott, M. Jamshidi, " Behaviour hierarchy for autonomous mobile robots: fuzzy-behaviour modulationa and evolution", International journal of intelligent automation and soft computing, Vol.3, No.1, pp. 37-49, 1997
[13] C.Voudouris, P.Chernett, C. Wang, V. Callaghan, “ Fuzzy hierarchical control for autonomous vehicles”, Proceedings of the international workshop on Intelligent Robotic Systems, 1994.
[14] .J. Zhang, V.Schwert, “ Rapid Learning of Sensor-Based Behaviours of Mobile Robots Based on B-Spline Fuzzy Controllers", 7th IEEE International Conference on fuzzy systems, Anchorage-U.S.A, 1998.