Monograph:
Zalzala, A.M.S. (1994) Explorations in Real-Time Robotics Using Parallel Processing, Neural Networks and Genetic Algorithms. Research Report. ACSE Research Report 507 . Department of Automatic Control and Systems Engineering

EXPLORATIONS IN REAL-TIME ROBOTICS USING PARALLEL PROCESSING, NEURAL NETWORKS AND GENETIC ALGORITHMS

A.M.S. Zalzala

Robotics Research Group,
Department of Automatic Control & Systems Engineering,
The University of Sheffield,
P.O.Box 600, Mappin Street, Sheffield S1 4DU, UK
EMail: rrg@sheffield.ac.uk

Research Report #507
March 1994

Abstract: This paper reports on recent research on advanced motion planning and control of articulated and mobile robotic systems. In addition to employing the principles of distributed and parallel processing to produce feasible real-time multi-processor networks, other new theories such as neural networks and genetic algorithms are deployed as possible solutions. All reported algorithms are implemented for either the PUMA 560 arm or the B12 mobile robot. The ultimate aim of the on-going research is to present working architectures for real-time robotic systems by augmenting all developed structures.

Keywords: Real-time robotic systems, distributed and parallel processing, artificial neural networks, genetic algorithms, transputers.

1 Introduction

Real-time robot control has always presented researchers with great difficulties in terms of both the accuracy of the command actions required and also the efficiency by which the commands are obtained. A very important characteristic of the new generation of robotic systems is the presence of intelligent capabilities which is being rapidly supported by fast computing power and adequate sensory equipment.

A general form of the robot control loop is shown in Figure (1), where the required job is first divided by the task planner producing a number of consecutive tasks, followed by the motion planner which gives a time history of positions, velocities and accelerations sufficient and necessary to realise each task. Once the desired motion elements are available, they are used to produce the commands for the individual joint loops via the control module which may or may not include the dynamic model of the system (model reference adaptive controllers vs. simple PIDs). The motion is realised by applying the control commands to the robot system and a feedback module provides the actual motion elements to cater for any uncertainties and/or changes in the system parameters and/or environment set-up. Overall, intelligence may be needed at different parts of the control loop, e.g. in connection with the task planner, dynamic model or sensory feedback.

Nonetheless, for complicated multi-joint mechanical chains, the above loop is very hard to accomplish in real-time and provisions must be made for efficient designs.

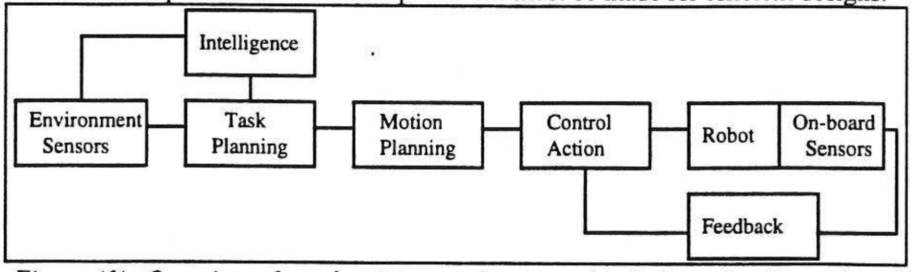

Figure (1): Overview of a robot integrated system: this is from the view point of an upper hierarchical structure and serving to illustrate the main modules; both the intelligence and feedback blocks can be deployed in other parts of the system as an implementation may require.

This paper will report on different architectures attempting to realise parts of the robot control loop, and further providing for intelligence in real-time applications. Three approaches are considered. First, medium-grain parallelism of the classical control approaches is investigated, presenting efficient structures on general purpose processing elements, where the transputer is chosen as a vehicle for the realisation of such structures. Second, fine-grain distribution of a module using artificial neural networks is tackled, where the massive parallelism and the learning abilities of the connectionist architectures is of vast interest. In addition, efficient search techniques for fast motion planning of multi-joint manipulators and mobile vehicles are enhanced by using genetic-based algorithms, hence providing a more optimum solution.

In addition to this introduction section, this contribution in divided into another three main sections each reporting on different algorithms using parallel processing, neural networks and genetic algorithms, respectively. In addition, a discussions and conclusions section is included.

2 Parallel processing structures on transputer networks

The complexity of the dynamic model of an articulated chain proved to be very computationally demanding for on-line applications. Thus, although simplifications have been proposed in terms of ignoring parts of the model, in particular at low speeds, it is a well established fact that such a practice seriously degrades the performance and tracking accuracy [1]. However, the relatively recent availability of low-cost general purpose processors such as the INMOS transputer has presented feasible candidates for the application of the principles of parallel and distributed processing in real-time operation. Nonetheless, due to the lack of any automatic means by which an algorithm can be recasted in a parallel form, extreme care must be taken to ensure high utilisation of all processing elements used within a network while keeping the overall system as cost-effective as possible. Thus, in applying the concepts of concurrency to design the parallel algorithm, two parameters are of overriding importance, namely the minimisation of the execution time and the maximisation of every processor utilisation, and using a time scheduling procedure is appropriate to maintain the latter requirements [2]. One further issue of interest is communication bottlenecks which must be eliminated to maximise efficiency.

This section reports on several parallel structures for real-time robot control. Although these structures can be applied using any general-purpose processor, the INMOS transputer is chosen as the main processor in all implemented networks due to the availability of a wide range of both software and hardware support. In addition, although recent benchmarks indicated the superiority of other processor architectures, such as the SPARC, in a stand alone mode, the transputer is still the best available for constructing multi-processing networks [3]. A final realisation of the overall integrated system is discussed in section 2.4.

2.1 Minimum-time motion planning

The minimum-time control of robot arms has usually been implemented as a two-task procedure, where the appropriate trajectory planning is performed off-line, and then tracking is carried out on-line to achieve the desired motion. However, such an approach is unacceptable whenever the motion is dependent on the on-board sensory equipment operated during the application. In on-line operation, the robot end-effector is required to track a path specified by the on-board sensors during motion. Therefore, the look-ahead point concept have been introduced [4,5] which requires the detection of a via-point to move to, planning the motion segment then tracking the intermediate points produced. While this present segment is traversed, the planner detects a next look-ahead via-point and plan the next segment for the robot to follow once traversing the first segment is completed. The construction of successive motion segments continues in real-time provided that the planner is fast enough to provide data to the manipulator, a highly critical requirement for high-speed automated applications. This concept of the on-line alternating between motion planning and trajectory tracking depends on real-time perception and execution thus illustrating an important aspect of intelligent sensory-based systems, where the robot will be able to work in dynamic and unstructured environments.

Nonetheless, for this concept to be implemented, fast computational modules, as well as fast perception abilities are required to maintain continuity of motion, hence the need for multi-processor networks. Algorithms are implemented for the six-joint PUMA 560 arm [6] where minimum-time motion planning in the configuration space is distributed at two distinct levels, as shown in Figure (2). In Figure (2), a global distribution is achieved by treating each joint of the manipulator separately while a finer local level of parallelism is shown by concurrently constructing and optimising different motion options for each joint. In addition, a control unit maintains the high operational performance of the overall structure. The complete parallel algorithm is implemented on a network of T805 processors as shown in Figure (3).

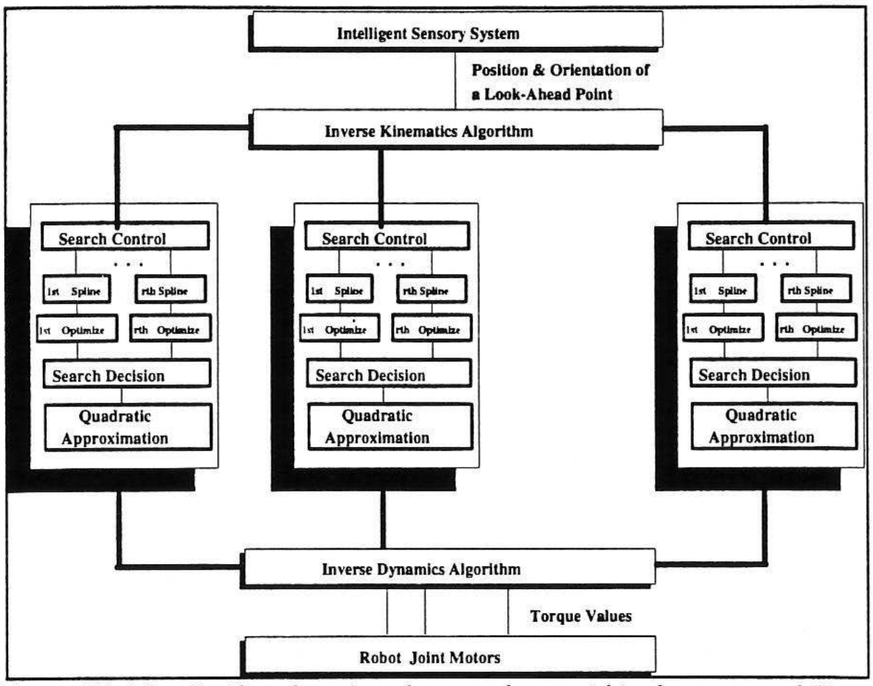

Figure (2): The distributed motion planner: shown within the context of Figure (1). A number of parallel modules are designated each for one joint of the robot arm. On each module, a number of concurrent processes are executed to spline, optimise and search for the minimum-time motion option.

2.2 Distributed dynamic equations of motion

The motion planner described in the above section provides the time history of positions, velocities and accelerations for the arm to follow. However, the control commands must be computed and downloaded to the joint loops for execution. Although simple single-joint P(I)D controllers are usually used, these are not efficient when high-speed motion is required or when changes in the robot model and/or the environment are encountered, and the dynamic equations of motion must be incorporated in the control module to provide for accurate tracking [7]. Nevertheless, the presence of these highly non-linear equations creates a very computationally expensive problem when combined with the requirements of minimum-time motion, despite several algorithmic simplifications.

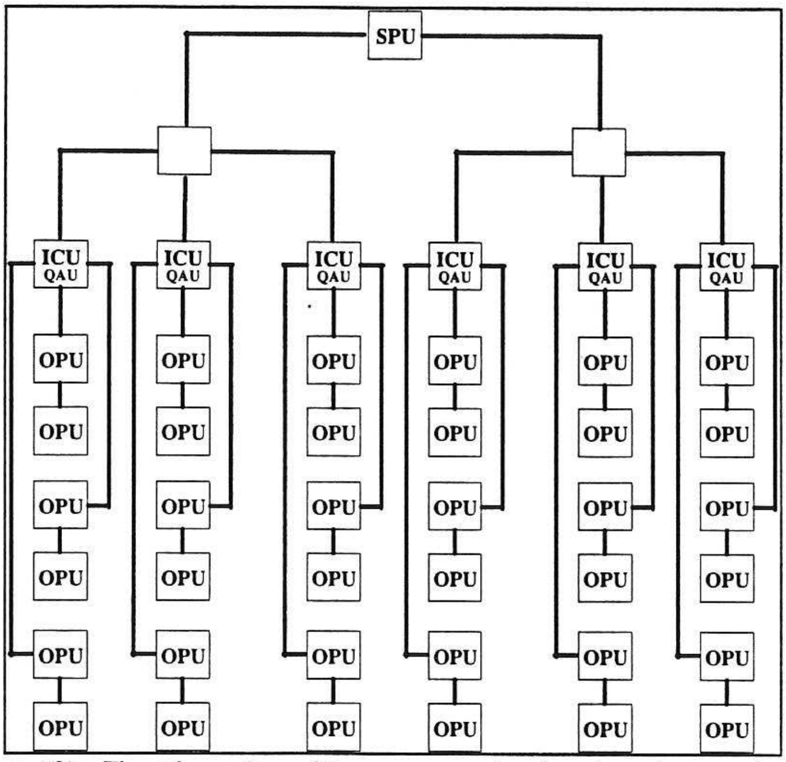

Figure (3): The planner's multi-processor network: six columns of seven processors are set for each joint of the PUMA. The intelligent control unit (ICU) runs different optimisation units (OPUs), gathers all possible options of motion and chooses the minimum-time one, which is in turn communicated to the supervisor unit (SPU). Note the limitation on communication links between the ICUs and their respective OPUs and also between the SPU and the six ICUs. The number of processors assigned to each joint can be modified depending on how fast the planner needs to operate in a particular application.

Hence, the efficient and fast computations of robot arm joint torques is considered, and a solution of the inverse dynamics problem through the design of a distributed architecture for the recursive Lagrangian equations of motion is presented [8]. This form of robot dynamics is a fully recursive formulation having a linear complexity of O(n) with two phases of recursion, backward and forward [9]. Parallelism is exploited at three distinct levels. First, a global level, where a pipelined configuration is imposed to accommodate for the presence of recursion in the equations. Second. a local level, where the procedure at each link is divided into several concurrent jobs. Finally, a sub-local level, yielding parallelism in the operations of a certain job within each link. In addition, a symbolic representation of parts of the relevant equations are developed as a contribution to the reduction of complexity yielding a high utilisation of each processing element. In implementing the algorithm, a two-dimensional torus array of processors is used with the vertical extension employed for the global recursive configuration, while the horizontal extension accommodates for the local distributions, as shown in Figure (4).

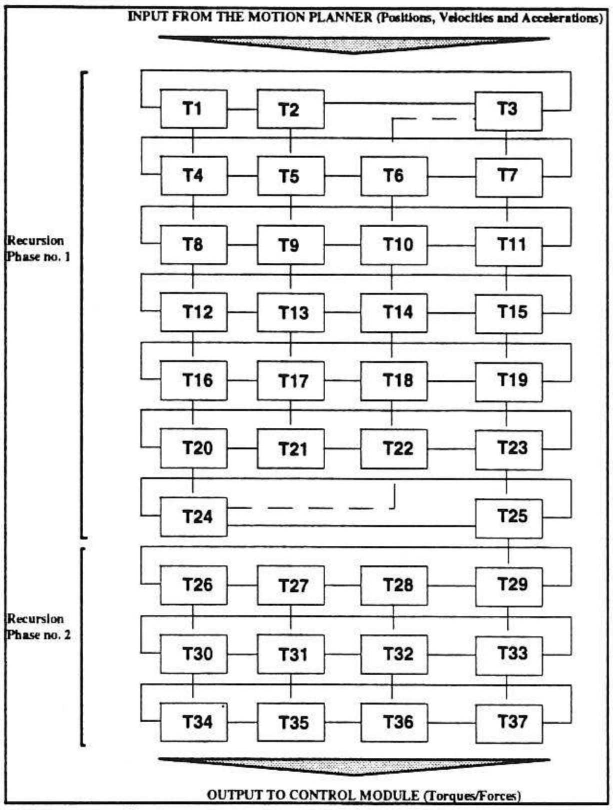

Figure (4): Torus multi-processor network for the dynamic equations: the recursion in both phases of the formulation is catered for via a pipeline shown vertically (7-stages in phase 1 and 3-stages in phase 2) while horizontal distribution is exploited within each stage of the pipeline. In all, 37 processors are needed for the six-joint PUMA.

2.3 Distributed measurements for the end-effector

As indicated in Figure (1), the robot must have some form of sensory equipment to provide a feedback on both the position and orientation of the end-effector. An efficient ultrasonic measurement device have been developed earlier [10] and further modified to present orientation information in addition to 3D Cartesian position [11]. Different hardware and software design issues have been reported in the literature, while emphasis here are made on the distributed implementation of the inverse kinematics transformations for the PUMA arm, where providing the feedback data must be fast enough to operate with the control module.

The experimental set-up shown in Figure (5) employs two sets of 8 ultrasound receivers (one at each corner of the volume) along with two sets of 5 transmitters located at the robot's hand, and the computations are carrier out using trigonometric relationships [12]. Two levels of concurrency are identified. First, a procedural level, operating the position detection, orientation detection and inverse kinematics as a pipeline producing the joint values. Then, a measurement level, where measuring the position and orientation of the hand is executed in parallel for all combinations of the receivers.

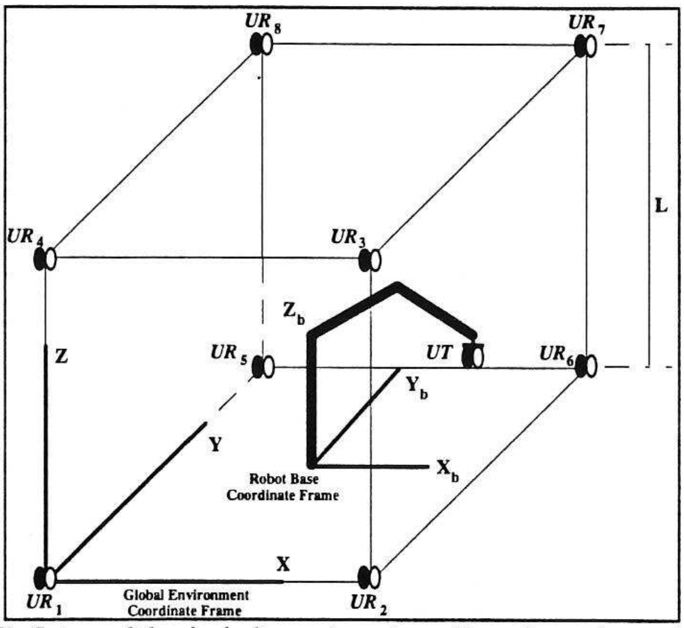

Figure (5): Set-up of the dual ultrasonic systems: Two ultrasonic receivers (UR) are placed in each corner of the volume (denoted as ● and O) along with two respective arrays of transmitters (UT) placed at the robot hand. By employing the measurements, both the position and orientation of the robot tip can be calculated. The eight-corner receivers structure also provides for fault tolerance by accommodating for transducer failure and overcoming problems associated with blocking of waves by obstacles in the environment.

The multi-processor system is shown in Figure (6) where eight concurrent nodes are employed, one for each corner of the volume, to measure and compute a number of samples, in addition to another four nodes forming a pipeline for computing the joint values. This architecture is tested on a network of ten transputers to be consistent with previous phases of the integrated system, although cost-effectiveness may be raised as an issue. However, the aim is to provide for an integrated system, as described in the following section.

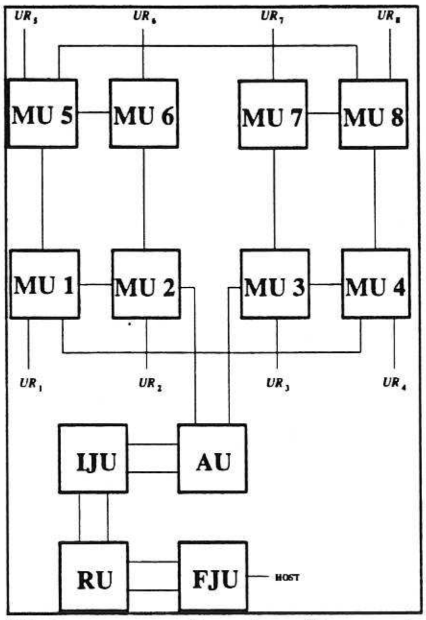

Figure (6): The multi-processor system for ultrasound measurements: eight measurement units (MU) are shown, providing data for an averaging unit (AU), an initial joint unit (IJU), a rotational unit (RU) and a final joint unit (FJU), all operating in a pipeline structure. The final output is a vector of six joint values for the arm.

2.4 <u>The integrated system</u>

The three distributed structures reported in sections 2.1, 2.2 and 2.3 are integrated together to present a massively parallel real-time robotic controller. This is shown in Figure (7) with a PD compensator added to facilitate the use of the computed torque values, while current work is progressing towards integrating an alternative adaptive structure [13].

The cost of VLSI structures are dropping continuously, and the above integrated system may not be very expensive considering that it can provide for a far better performance for an automated system which may cost excessively more than the controller. In addition, further modules must be added to enhance the system, in particular a more sophisticated adaptive controller and a high-level sensory architecture.

3 The learning control of dynamic systems

The neural network structure used by engineers is generally modelled on the biological nervous system and the brain in particular, where it is hoped to inherent the latter's ability to learn and perform fast computations. Nonetheless, due to the limited knowledge possessed of the brain structure and functions [14], the mathematical model of neurones is more of an ad-hoc approach rather than a well organised one. Unfortunately, neural networks formed a fashionable field to follow by many engineering researchers while still uncertain of the ability of the theory of cognition to present a suitable replacement of the existing well-established control theory. Although proving successful in many applications when used as a classifier, the use of an assumed form of neural networks in real-time control of dynamical systems proved a failure. This failure resulted mainly from the fact that an ad-hoc structure of a multi-layered network could not be generalised to accommodate for all operational modes of a non-linear dynamic system, although showing some good results in individual examples.

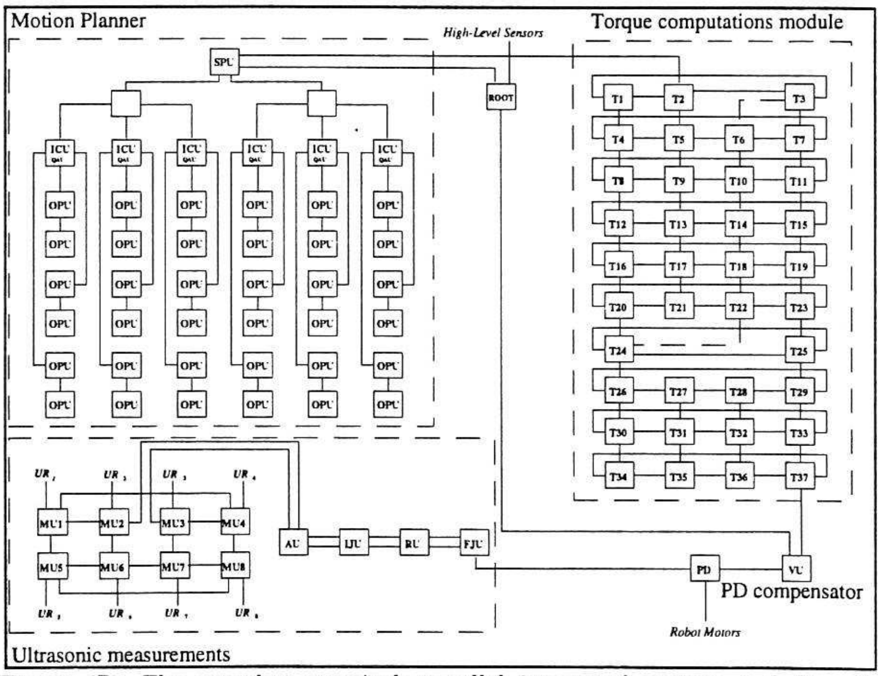

Figure (7): The complete massively-parallel integrated system: including the motion planner, the torque/force computations module, the measurement module and a PD compensator.

A novel approach to neural robot control was developed, where conventional robot control theory was augmented with aspects of the theory of cognition [15]. This approach gives the neural network some initial knowledge of the system model, where the network is expected to behave better once it knows something about its task. Such initial knowledge can be readily available in engineering applications where data of the controlled system is provided for, as it is the case in robotics. In the case of a robot arm, this initial knowledge is presented by including part of the model as the combining function on each of the neurones. The main aim of this approach is to make use of the parallelism for real-time implementations, in addition to the learning abilities to perform adaptive control. However, no model identification is attempted here since the robot model is already defined, while the remaining task is efficiently controlling the arm.

Implementations are accomplished for both the kinematics and dynamics formulations of the full model of the PUMA 560 arm [16], hence illustrating the practicality of the procedure for any robot structure with any number of joints. The complexity of the system can be demonstrated in Figure (8) showing the neural structure for the velocity term of the Lagrangian equations of motion. In addition, Figure (9) shows the network for the inverse Jacobean. It must be emphasised that the network structures are representatives of the model complexity as both the number of hidden layers and the number of neurones in each are determined by the mathematical model and not assumed a priori. Although this implementation is reported here for a robotic system, the same procedure can be applied to solve any problem where real-time dynamic systems are involved.

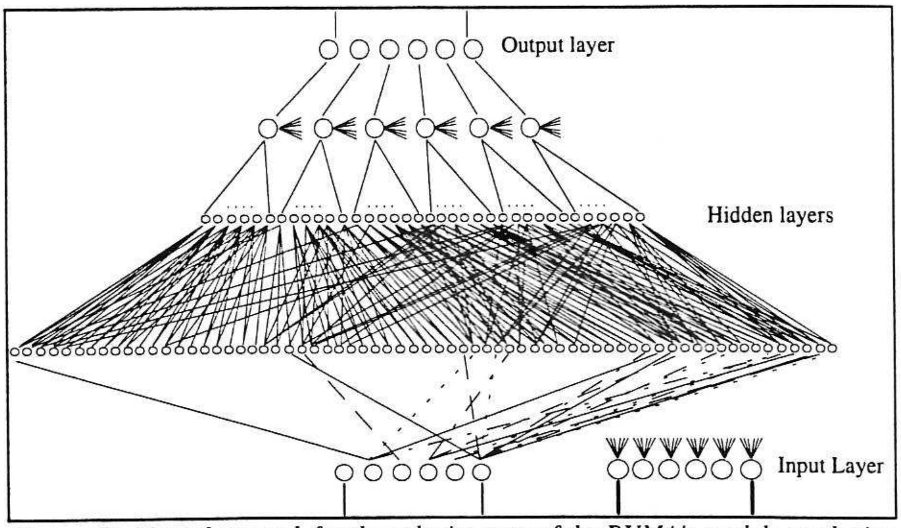

Figure (8): Neural network for the velocity-term of the PUMA's model: employing two hidden layers where the mathematical model is distributed over a total of 103 neurones. The number of layers, number of nodes and interconnections are imposed by the dynamic formulation while a modified back-propagation algorithm is used to accommodate for the changes.

3 Genetic-based motion planning

Genetic algorithms are adaptive search techniques mimicking the process of evolution by emulating the concept of the survival of the fittest. Due to different attractive features [17], genetic-based procedures attracted much interest in many fields including robotics. This section is concerned with introducing genetic-based algorithms for the minimum-time trajectory planning of articulated and mobile robotic systems. Although heuristic search techniques are available and can be implemented on parallel structures, as reported in section 2.1, the fact that an infinite number of solutions exist to move from one point to another renders their operation as sub-optimal [18] and becomes even harder to achieve in real-time when a large working space is considered.

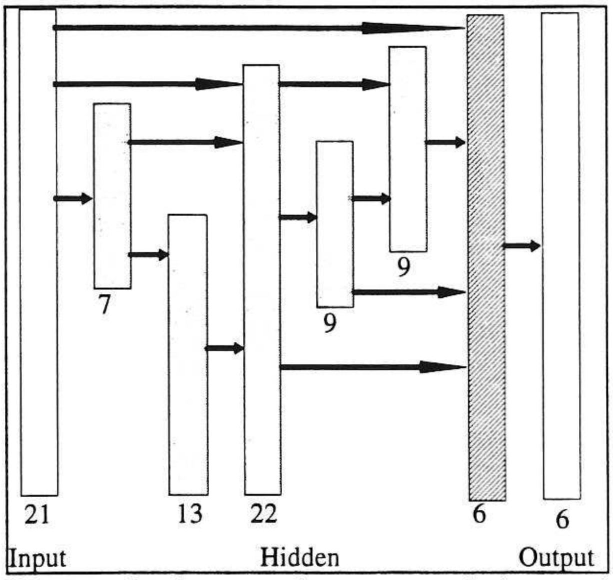

Figure (9): The inverse Jacobean neural structure: employing a total of six hidden layers for which the number of nodes are shown. Note in particular the direct interaction between the third and sixth hidden layers and also the direct input to the sixth hidden layer. The shown structure is, again, imposed by the mathematical model.

In using a genetic-based approach, the main aim is to provide an efficient planning algorithm in terms of execution time while respecting the limitations imposed by the design. For an articulated arm, the planning procedure is performed in the configuration space and respects all physical constraints imposed on the manipulator design including the limits on the torque values applied to the motor of each joint of the arm [19]. Consequently, the complete non-linear dynamic robot model is incorporated in the formulation. This feasible algorithm emerged through an evolutionary process while taking account of different concerns related to robot motion planning, and the binary representation of the chromosomes is illustrated in Figure (10). Reproduction was controlled to prevent premature convergence, the analogous crossover directed sensible crossover operations, and specially shaped mutation operators promoted new search space. The algorithm is proven to be far more efficient compared to the conventional heuristic search techniques with a reduction in the execution time of 1:20. Although results are presented for a simple two-link manipulator, the algorithm is readily applicable for any number of degrees-of-freedom.

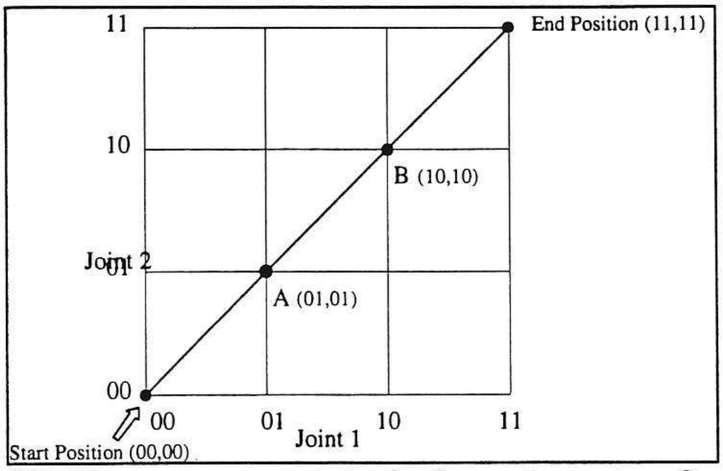

Figure (10): Chromosome representation for the genetic structure: Considering 2D space, a joint position is represented in binary form as shown, while a representation of 00,01,10 and 11 are used to indicate motor saturation in either direction for both joints, respectively. Hence, moving from the starting point to the end point via A and B is represented as {0101\01; 1010\10} with the joints' motor commands saturated at the negative and positive bounds, respectively.

A similar situation, though rather simpler, arises when planning the motion of mobile vehicles. Although many solutions may exist, a condition for obtaining the best (or near best) option may be imposed by the user, where a criterion in terms of the total distance traversed, energy expended or minimum execution time must be achieved. The planning procedure is made more complicated if the robot has to detect and avoid static or dynamic objects in the workcell.

Genetic algorithms are deployed to investigates a more detailed and sophisticated approach to planning in 3D space involving moving (or disappearing) obstacles, thus accommodating for the concept of intelligent control within a dynamic environment. Prior to the planning process, a global knowledge of the environment is needed and is translated in the form of a terrain map. Firstly, a set of valid random paths are generated as the initial generation. In order to prevent the robot wandering endlessly inside the workcell, a weighted vector of motion is employed during the path construction phase, that is, the direction of motion from the robot's current position towards the goal has more chance to be chosen than other directions. A fitness value is assigned to each path and the one with the best fitness is stored for future usage. Secondly, pairs of paths are chosen randomly for mating, and those with better fitness will have more chance to be drawn out for mating. The reproduction process repeats until some arbitrary number of generations is reached [20]. To realise the results obtained via the simulation, an actual implementation utilising the B12 mobile robot was accomplished, where maps stored through the on-board sonar array are manipulated by the genetic planner to present a feasible motion. Figure (11) illustrates the concept of dynamic environments, while Figure (12) shows the genetic planner in operation.

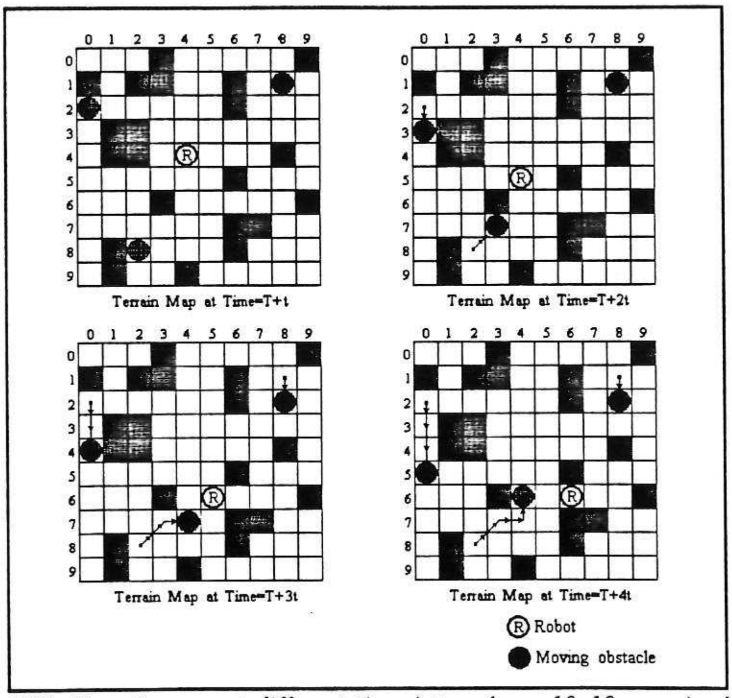

Figure (11): Terrain map at different time intervals: a 10x10 map is shown at different time intervals while the planner manoeuvres the robot to avoid a moving obstacle. This procedure may be easy enough to accomplish in simulation, however, a practical system will require updating the map every time interval via data obtained from the sonars where real-time issues must be addressed.

4 Discussion and conclusions

This paper reported on different robotic control structure using parallel processing, neural networks and genetic algorithms, with emphasis on real-time operation. The use of distributed processing, whether fine-grain or medium-grain, is deemed as essential to realise an intelligent machine working on-line. The use of new theories such as connectionist and genetics is seen as a feasible development within the framework of the already existing and established control theory. Nonetheless, the researchers' findings lead to the conclusion that although the idea of creating a human brain is both interesting and exciting, a cautious and calculated approach must be followed to ensure a working design. Although an ad-hoc approach may yield some satisfaction in limited examples, it will never be appreciated for practical systems let alone competing with the classical theory.

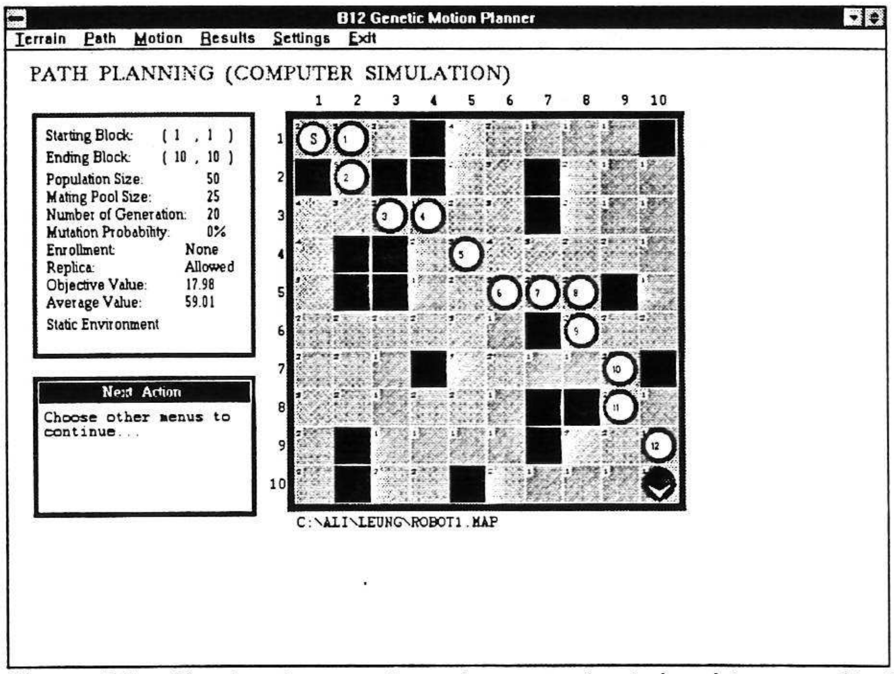

Figure (12): Planning in a static environment: A window-driven graphical interface is designed to monitor the on-line operation of the genetic-driven B12 mobile robot. The size of the terrain is adjustable according to the application and is built via data obtained from the B12's sonars. System can operate in both manual or auto-detection modes. Analysis of the motion results are stores and can be accessed by the operator via the Results menu.

The design of multi-processing systems utilising general purpose processors is more appreciated where implementation of practical systems are concerned, as these networks are available in VLSI. However, the efficient utilisation of neural structures is still hampered by the lack of a general purpose chip capable of accommodating for the designed algorithms. This is of particular significance when considering any integration between a neural processor and general-purpose processors. In this pioneering stage of the research, genetic-based approaches to robot motion planning show great potential where the results suggest that their performance may be substantially improved with further work.

Acknowledgements The author acknowledges the useful discussions and contributions by colleagues including Alan Morris, Kwong Chan, David Leung, Mike Dickinson, Ian Cornish and Fatollah Shahidi. Parts of this work are funded by Sheffield University Research Fund.

References

[1] J. M. Hollerbach, Dynamic Scaling of Manipulator Trajectories, Trans. ASME, J. of Dyn. Syst., Meas. and Control , 106, 102-06, 1984.
[2] H. Kasahara and S. Narita, Practical multiprocessor scheduling algorithms for efficient parallel processing, IEEE Trans. Computers, Vol.33, No.11, 1984.
[3] Cornish, I.J. and Zalzala, A.M.S.,"Transputer benchmarks and performance comparisons with other computing machines", In Proc. Workshop on Parallel Processing in Education, pp. 138-143, University of Miskolc, Hungary, March 1993.
[4] Zalzala, A.M.S. and Morris, A.S., "A distributed on-line trajectory generator for intelligent sensory-based manipulators", *Robotica*, Vol. 9, No. 2, pp. 145-55, 1991.
[5] R. H. Castain, R. P. Paul, An On-Line Dynamic Trajectory Generator, Int. J. Robotics Research, Vol.3, No.1, pp.68-72, 1984.
[6] Zalzala, A.M.S. and Morris, A.S., "A distributed robot control on a transputer network", *IEE Proceedings, Part-E*, Vol. 138, No. 4, pp. 169-76, 1991.
[7] J.J.E. Slotine and N. Gunter, "Performance in adaptive manipulator control", Int. J. Robotics Research, Vol. 10, No. 2, 1991.
[8] Zalzala, A.M.S. and Morris, A.S., "A distributed pipelined architecture of robot dynamics with VLSI implementation", *Int. J. Robotics and Automation*, Vol. 6, No. 3, 1991.
[9] J. M. Hollerbach, A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity, IEEE Trans. Syst., Man, Cyber., SMC-10, 730-36, 1980.
[10] M. Dickinson, A. S. Morris, Co-ordinate Determination and Performance Analysis for Robot Manipulators and Guided Vehicles, IEE Proceedings, Part-A, 135, 2, 95-98, 1988.
[11] Zalzala, A.M.S., Ultrasonic location measurement for fast robot control, Research Report no. 380, AC&SE Dept., Sheffield University, 1990.
[12] Zalzala, A.M.S. and Morris, A.S., "Fast ultrasonic location measurement for robot control", In Proc. Canadian Conf. on Industrial Automation, Vol. 1, pp. 15.23-15.26, Montreal, 1992.
[13] Ziauddin, S.M. and Zalzala, A.M.S., "Parallel processing for real-time adaptive control: Theoretical issues and practical implementation", In IEE Colloquium on High performance applications of parallel architectures, 1994

[14] F. H. C. Crick, C. Asanuma, Certain Aspects of the Anatomy and Physiology of the Cerebral Cortex, In Parallel Distributed Processing : Explorations in the Microstructure of Cognition, Eds.: D. E. Rumelhart and J. L. McClelland, Vol.2, pp.333-40, MIT Press, 1986.
[15] Zalzala, A.M.S. and Morris, A.S., "A neural networks approach to adaptive robot control", *Int. J. Neural Networks*, Vol. 2, No. 1, pp. 17-35, 1991.
[16] Zalzala, A.M.S., "A neural network methodology for engineering applications: The learning control of robotic systems", in *Neural Computing: Research and Applications*, Ed. G. Orchard, Institute of Physics Pub., 1993.
[17] D. E. Goldberg, *Genetic algorithms in search, optimisation and machine learning*, Addison Wesley Pub., 1989.
[18] G. Sahar, J. M. Hollerbach, Planning of Minimum-time Trajectories for Robot Arms, Int. J. Robotics Research, 5, 3, 90-100, 1986.
[19] Chan, K.K. and Zalzala, A.M.S., "Genetic-based minimum-time trajectory planning of articulated manipulators with torque constraints", In IEE Colloquium on Genetic algorithms for control systems engineering, May 1993.
[20] Leung, C.H. and Zalzala, A.M.S., "A Genetic Solution for the Control of Wheeled Robotic Systems in Dynamic Environments", Proc. IEE Int. Conf. Control'94, 1994.