Monograph:

Zalzala, A.M.S., Dodds, G.I. and Irwin, G.W. (1993) Advanced Control Structures: A Transputer-Based Multi-Robot System. Research Report. ACSE Research Report 468 . Department of Automatic Control and Systems Engineering

ADVANCED CONTROL STRUCTURES: A TRANSPUTER-BASED MULTI-ROBOT SYSTEM

Ali M.S. Zalzala, Gordon I. Dodds and George W. Irwin

Research Report #468

February 1993

ADVANCED CONTROL STRUCTURES: A TRANSPUTER-BASED MULTI-ROBOT SYSTEM

Ali M.S. Zalzala [1], Gordon I. Dodds [2] and George W. Irwin [2]

[1] *Robotics Research Group, Department of Automatic Control and Systems Engineering, The University of Sheffield, P.O. Box 600, Mappin Street, Sheffield S1 4DU, United Kingdom*
Email: ali@uk.ac.sheffield

[2] *Control Engineering Research Group, Department of Electrical and Electronic Engineering, The Queen's University, Stranmillis Road, Belfast BT9 5AH, United Kingdom*

ABSTRACT

This paper presents the implementation of an efficient motion planner for a two-robot system operating in a unified environment. The distributed formulation of the planner is mapped on a network of T800 transputers and caters for the needs of minimum-time and collision free robot motion in real-time. Practical implementation of the planner is reported for two co-ordinated RTX robots, and performance results are given of an execution run.

1. INTRODUCTION

The concepts of multiple robots operating in a common environment is very attractive to industry where increased productivity is required. Therefore, the design and implementation of advanced control structures for such environments has been high on the agenda of research and development institutions over the last few years.

Implementing a multi-robot system incorporates many complexities ranging between high-level sensory based data detection and processing, general task planning and co-ordinated motion planning, and lower level motion tracking and individual joint control. All the above tasks, in particular co-ordinated motion planning, require substantial computational performance in real-time [1].

Parallel processing and distributed algorithms have been identified as a potential approach to achieving real-time performance for the above systems [2] supported by the rapid availability of computer architectures, such as the INMOS transputer, which can be used as a basic element in the multi-processor network. Nonetheless, the issue of recasting the algorithms in an efficient distributed formulation remains open, where different approaches can be followed ranging between fine-parallelism (e.g. systolic arrays and neural networks) and gross-parallelism (e.g. transputer networks), and including a mixture of both.

This paper reports on the implementation of a multi-robot motion planner on a network of 6 transputers. Considering a functional decomposition of the mathematical procedures, an efficient minimum-time and collision-free distributed planning algorithm is produced. Due to the very heavy computational burdens inherent within the algorithm, the gross-parallelism approach is seen as appropriate for implementing the system on the available transputers.

2. The multi-robot motion planner

The motion planner consists of two main procedures, where the first procedure supplies minimum time motion (MTM) history for both robots while the second procedure ensures a collision free (CF) path exists in the shared environment to execute the required task.

2.1 Functional decomposition of procedures

The planning algorithm works as follows. Starting from rest, the system detects a first via-point for both robots to move to, and the MTM planning of both robot motions is performed simultaneously, providing a level of parallelism. Once time histories of possible motions are available for both robots, the best CF motion is detected and sent to the controller to execute. The MTM procedure used treats every joint on each arm individually, thus providing for another level of parallelism. The CF procedure is only activated after the MTM terminates and checks different motion options concurrently.

The above functional decomposition is illustrated in Figure (1). In Figure (1), the planning for both robots is computed in parallel, where the MTM is activated first for all n joints of each arm producing different motion options. Then, the CF is activated checking all possible options presented by the MTM and choosing an optimum one. Nonetheless, certain communications need to take place between the two robot modules, as described in the following section. A further insight into the mathematical formulations of different MTM and CF procedures is reported earlier [3,4] while this paper concentrates on the implementation.

3. Mapping on the transputers

The decomposed procedures of Figure (1) are mapped efficiently on a network of 6 T800 transputers which are available for this project. The programmes were adopted for two RTX robot arms with 6 joints on each. Initially, three transputers are reserved for each robot, with the MTM and CF modules operating successively. Thus, the planning will be performed in two phases.

1. *Phase 1:* Each transputer, in a set of three associated with a robot, computes all possible minimum-time motion (MTM) segments of two joints of the robot. Then, all options of all six joints are stored one transputer, and all six MTM modules are terminated.
2. *Phase 2:* The two transputers holding the total MTM results of both robots exchange the data. Thus, sets of multi-robot motion options will be checked successively on all six transputers, until a CF option is found. Once a feasible motion option is detected, all six CF modules are terminated and the choice sent back to the supervisory control module.

The mapping of all MTM and CF modules on the six-transputer network is shown in Figure (2). Considering the structure described above and making use of the transputer abilities to run simultaneous tasks, up to four separate modules were initiated simultaneously on a single transputer. However, as only one module is activated at a time, no resources sharing is encountered.

To illustrate the system operation, consider the modules on the master transputer shown in Figure (2). The Control module initiates the MTM module on its transputer and subsequently all other processors then become idle. Once the MTM module receives all data, it communicates it back to Control and halts. Control then activates the CF module and waits for its results. Once CF transmits the data and halts, Control passes it to the host for execution. For the overall software structure, a total of 19 modules are running in parallel on 6 processors. To facilitate communication between different sets of MTM and CF modules, a multiplexer/demultiplexer software module is present on each transputer.

4. PRACTICAL IMPLEMENTATION

4.1 THE COMPLETE SYSTEM

The motion time history produced by the CF-MTM network is downloaded to the actual robot's joints via an interface system. The hardware/software interface system is implemented on a network of 4 T414 transputers and is reported earlier in the literature [5], along with the software adaptation to augment it with the CFMTM network [6]. Both the planning network (CFMTM) consisting of six T800s and the interface network (RTXLINK) consisting of four T414s are programmed in parallel ANSI-C and hosted by a Sun SPARC-IPC workstation as shown in Figure (3). The X-windows environment running under Unix provides a powerful tool for handling the transputer networks.

When the multi-arm systems control script is activated from within the X-windows environment, two shells are executed creating two windows one each for the RTXLINK and the CFMTM networks, and downloading the appropriate executable codes. The RTXLINK initiates the two robots and starts polling the CFMTM requesting motion data. Once the CFMTM is ready it communicates the planned motion to the RTXLINK interface to execute it, while the CFMTM goes back to planning another segment. Interaction between the two networks is maintained via flag data files. The windows-based multi-robot environment is shown in Figure (4).

4.1 RESULTS OF A CASE STUDY

The multi-arm motion planner was tested for a sample motion, where a comparison between execution times on a single transputer and on the network is reported in Table (1). From Table (1), the CF is shown to impose a more intense computational burden due to the vast computations involved. The distributed formulation shows a reduction ratio in extection time of about 1:6 when compared to the sequential algorithm.

Module	1 Transputer	6 Transputers
MTM	4.32	1.02
CF	295.80	47.09
Total	300.12	48.11

Table (1): Execution Times Comparison (seconds)

5. FUTURE WORK

This practical system can be used as a development platform for augmenting other distributed structures of different related problems, as described in section 1. Efficient transputer implementation of robot dynamics and Kinematics as well as sophisticated adaptive control schemes [7] can be plugged on and tested. One further research possibility is augmenting the gross-parallelism approach used with a fine-parallelism algorithm implemented by a neural structure. Although efficient simulations can be found [8], a realisation of such a hybrid system is rather difficult due to practical implementation difficulties.

6. Conclusions

The complexity of planning a co-ordinated motion of two robot manipulators in real-time is overcomed in this paper by introducing an efficient distributed formulation on a network of 6 processors. In addition, a practical implementation of the algorithm is reported for two RTX manipulators working in a common environment. The total system hardware comprises a network of 10 transputers hosted by a Unix-based SPARC-IPC workstation, where in addition to the planning algorithm reported here, an additional 4 transputers are used to interface to the two robots. Making use of the ability of the Unix operating system to initiate different shells within its window environment, both the planning and interface networks are activated simultaneously by downloading the appropriate booting codes. The system is seen as a development tool for further phases of advanced robotics research.

Acknowledgement

The authors acknowledge the support of the Technology Board for Northern Ireland (Grant No. ST79).

References

1. Koivo, A.J. and Beckey, G.A., 'Report of workshop on coordinated multiple robot manipulators: planning, control and applications', IEEE J. Robotics and Automation, Vol. 4, No. 1, pp. 91-3, 1988.
2. J. H. Graham, High Performance Computer Architectures for Robotics and Automation, Computer Architectures for Robotics and Automation, J. H. Graham, Gordon and Breach Science Pub.,pp. 1-19, 1987.
3. Zalzala, A.M.S. and Morris, A.S., 'A distributed robot controller on a transputer network', IEE Proceedings, Part E, Vol. 138, No. 4, pp. 169-76, 1991.
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. Wilson, G., Irwin, G.W. and Dodds, G.I., 'Development of a transputer based hardware/software system for the control of a SCARA robot', In Proc. Int. Conf. on Parallel Computing and Transputer Applications, Barcelona, Spain, 1992.
6. Zalzala, A.M.S. and Dodds, G.I., 'The QUB multi-arm system - algorithm description', Internal report, Dept. of Electrical and Electronic Engg., The Queen's Univ. of Belfast, Sep 1992.
7. 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, pp. 117-28, 1991.
8. Sinha, P.K. and Ho, P.-L., 'Transputer-based real-time parallel adaptive control of a robot manipulator', in Proc. IMACS/IFAC Int. Symp. on Parallel and Distributed Computing in Engineering Systems, pp 187-92, 1992.

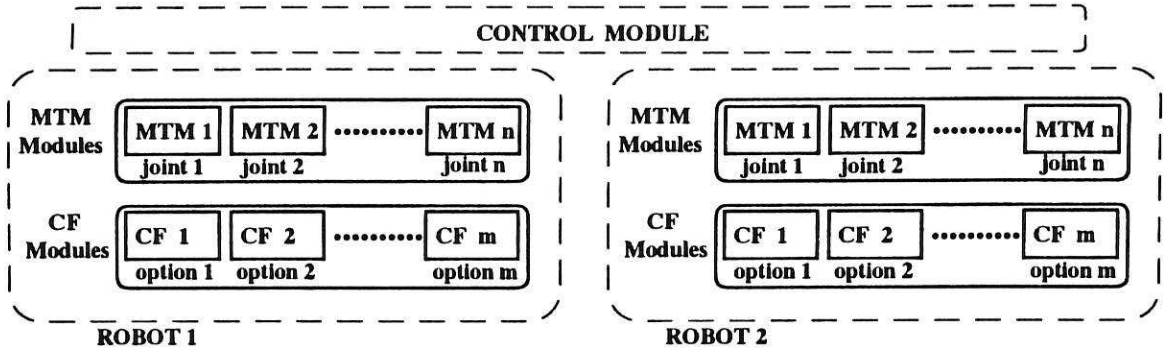

Figure (1): Functional Decomposition of Procedures

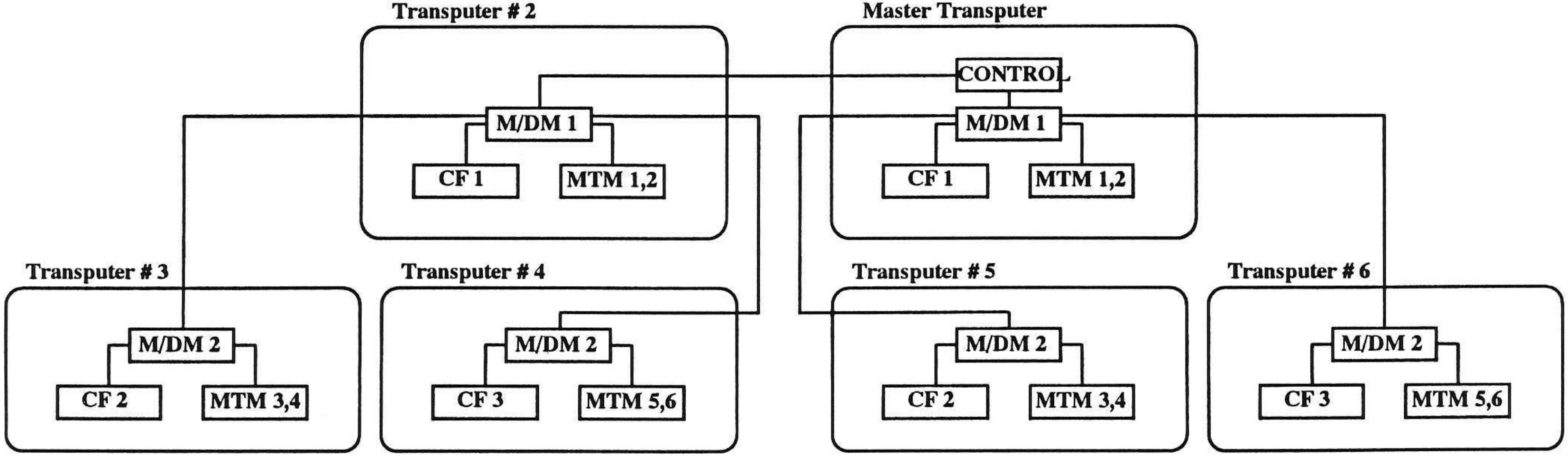

Figure (2): CF-MTM Configuration on the Transputer Network

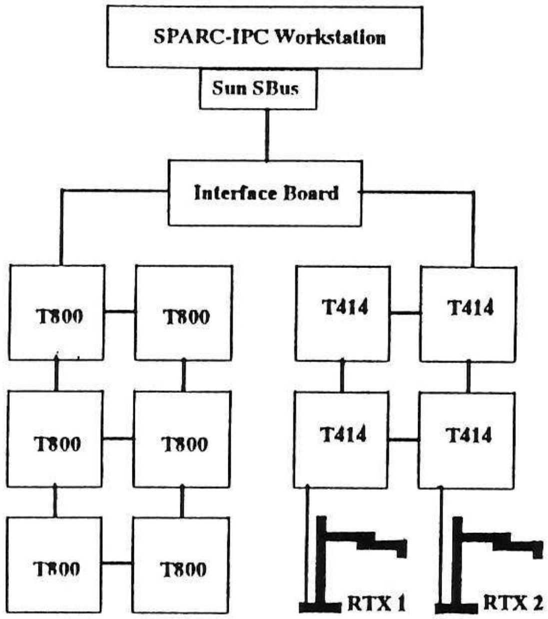

Figure (3): The Complete Practical System

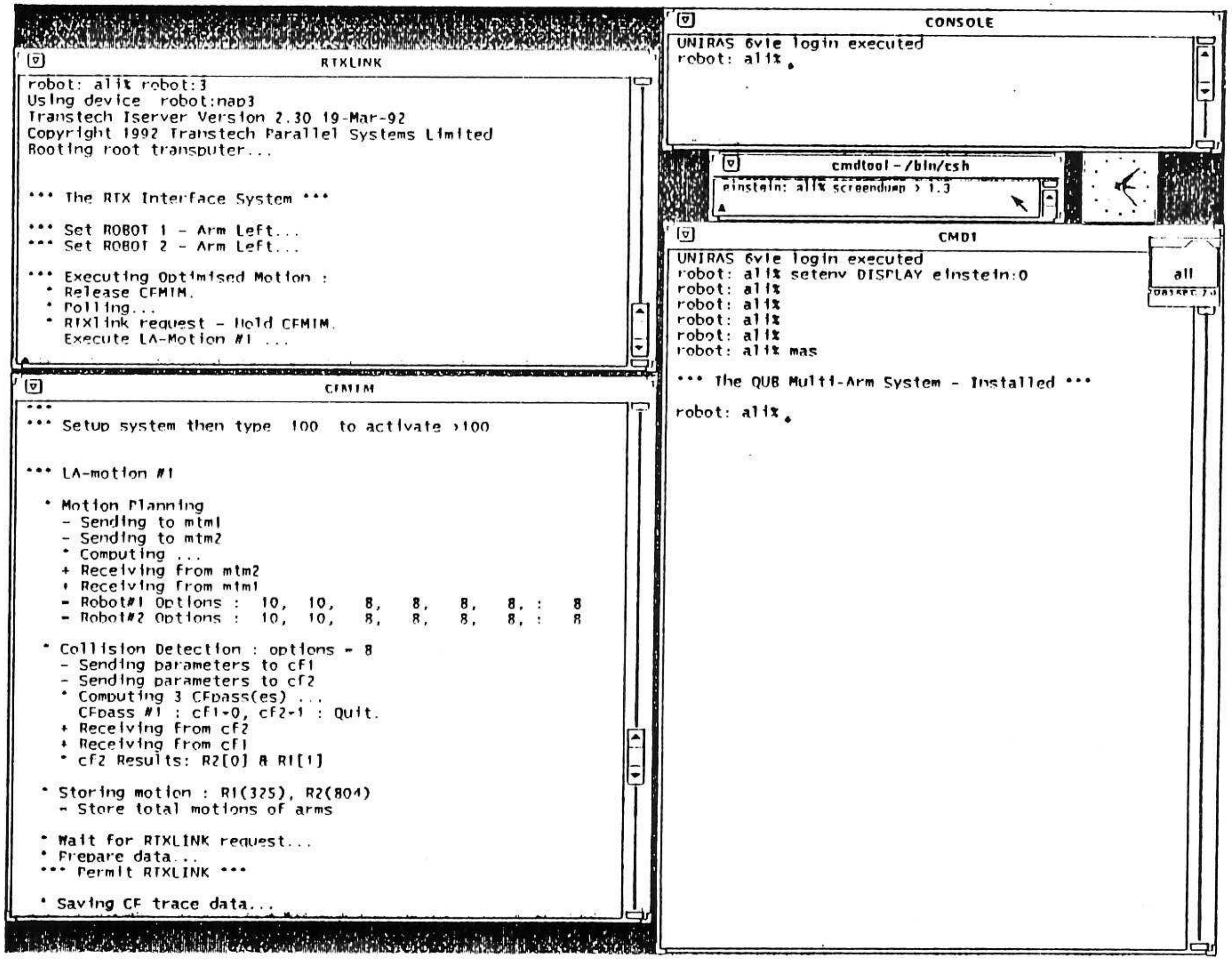

Figure (4): The Windows-Based Environment