This is a repository copy of *A High-Performance Multi-Arm Environment: Theoretical Aspects and Practical Implementation*.

White Rose Research Online URL for this paper:
http://eprints.whiterose.ac.uk/79351/

Monograph:
Zalzala, A.M.S., Dodds, G.I. and Irwin, G.W. (1993) A High-Performance Multi-Arm Environment: Theoretical Aspects and Practical Implementation. Research Report. ACSE Research Report 479 . Department of Automatic Control and Systems Engineering

Reuse
Unless indicated otherwise, fulltext items are protected by copyright with all rights reserved. The copyright exception in section 29 of the Copyright, Designs and Patents Act 1988 allows the making of a single copy solely for the purpose of non-commercial research or private study within the limits of fair dealing. The publisher or other rights-holder may allow further reproduction and re-use of this version - refer to the White Rose Research Online record for this item. Where records identify the publisher as the copyright holder, users can verify any specific terms of use on the publisher's website.

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk/

629.8(S)
PAM BOX
X

A HIGH-PERFORMANCE MULTI-ARM ENVIRONMENT: THEORETICAL ASPECTS AND PRACTICAL IMPLEMENTATION

A.M.S. Zalzala[1], G.I. Dodds[2] and G.W. Irwin[2]

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

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

Research Report #479
May 1993

A HIGH-PERFORMANCE MULTI-ARM ENVIRONMENT: THEORETICAL ASPECTS AND PRACTICAL IMPLEMENTATION

Abstract*

This paper reports on the development of a multi-arm robotic system, where the practical implementation of the system involves motion co-ordination of two multi-joint RTX robots with six degrees-of-freedom, hosted by a SPARC-IPC workstation. A planning scheme is introduced to provide accurate and co-ordinated collision-free motion. In addition, taking into account the need for high productivity in industrial environments, minimum-time movements are imposed by increasing the manipulators' performance to a maximum, thus providing a high-performance workcell. A real-time case study is included to show the validity and efficiency of the system.

SHEFFIELD UNIV. LIBRARY ST. GEORGE'S

* This work was carried out while Dr. A.M.S. Zalzala was with The Queen's University of Belfast.
The authors acknowlage the support of The Technology Board for Northern Ireland (Grant no. ST79).

1. Introduction

Robot manipulators have spread widely throughout the industrial community as a result of the increasing need for greater productivity and higher quality end-products. Nevertheless, the use of a single-robot system in an automated manufacturing environment can prove inefficient where the tasks required by a particular procedure can be too complicated for a single manipulator to perform (e.g. assembly tasks, deep-sea exploration) [1]. One solution to this need is a *cooperating multi-arm system*, where two (or more) arms perform certain sub-tasks of the required procedure and, by being able to perform different complex tasks within the workcell, automated productivity is increased.

Although the use of cooperating multiple-arms is seen as essential for wider robotic useage by the industrial community, their actual implementation proves to be complicated. Research in the design and operation of multi-robot cells has been going on for a number of years, and the problems related to the control of such a cell are many times more difficult than a single-arm system [1,2].

The increased interest in multi-robot systems is evident by the large volume of research reported in the literature. The majority of reports is concerned with cooperative multi-robots, where the interaction between the arms is addressed [3,4,5] including the dynamic control of closed chains [6,7] and when used on a mobile platform [8]. In addition, two approaches to coordinated multi-robots are reported. In the first approach, only one arm is allowed to use the common work space at a time [9] which leads to delays in executing the task. The other approach is the planning of a collision-free path for all arms sharing the environment using complex optimisation and search techniques [2,10,11] and introducing new concepts such as neural networks [12]. However, all the above contributions stop short of actually implementing the proposed systems.

This paper describes the formulation and implementation of a coordinated multi-arm system (MAS). A task controller is set up for coordinating the movements of two robot manipulators, and for ensuring collision-free (CF) motion. The method used for collision detection exploits maximum flexibility in movements, allowing the manipulator to make the best use of the available space. For each manipulator a trajectory planner is designed, giving the minimum-time motion (MTM) while taking into account the physical constraints imposed by the arm. In addition, employing a search technique within the MTM planner provides other, less optimal, options to use in case the minimum-time decision is not feasible. Eventually, the *near optimum and possible* motion is specified by which the cell can perform its required function efficiently. The MAS software is written in C on a SPARC-IPC workstation and results are reported for the actual implementation on two RTX manipulators with six-degrees-of-freedom. A sample case study is reported in this paper, where 3-D plots of the movements of the arms indicates the common areas of intersection between originally assumed trajectories, along with the newly defined feasible motion.

This work is presented as follows: in section 2 an overview of multi-arm systems is given, where all relevant control modules are discussed. Section 3 includes the theoretical aspects behind the system, while its practical implementation is described in section 4. Case study results are reported in section 5, showing the validity and efficiency of the system. In section 6 some ideas are given of the on-going development of the system, while section 7

concludes this paper.

2. Multi-Arm Workcell Requirements

Multi-arm robot control involves two distinct and equally important aspects, namely: (1) motion coordination within a common workspace and (2) interaction between arms when handling objects. Figure (1) shows the different control modules required in approaching the above aspects.

The first aspect is concerned with organising the general robot motion within the shared environment, which includes *task planning* (determining the required tasks for all arms to perform), and *trajectory planning* (producing the time history of motion for all arms). In addition, collision avoidance must be addressed to ensure the possibility of completing the required job, since collision can be expected between the robots limbs as well as with other obstacles in the workspace. This second aspect involves interaction in the way the planned motion is executed. Thus, the *trajectory tracking* module, which generates the commands for the motors employing a certain control law, will be different. This point will be illustrated in the following subsections which explain the different modules mentioned above.

2.1. Task Planning

Motion strategy decisions are at the top of the control hierarchy. Thus, the task planner must be able to decompose the main task into sub-tasks, each carried out by one of the cooperating arms, and further give instructions to the relevant robot motion planners on how to go about performing the task. As shown in figure (1), the completion of the task planning involves acquiring information on both the environment and the current state of each of the arms. This involves the automatic processing and analysis of data provided by different integrated sensors, while a more sophisticated system may involve intelligent capabilities as well. Consequently, for each robot arm, a set of *via-points* are provided, of which the location and frequency is dependent on the task required; a closer set of points may be required to execute a more precise and finer motion. Thus, depending on which approach the completion of the task requires, the planner will provide the appropriate motion strategy.

2.2. Motion Planning

This module is concerned with providing a time-history of intermediate configurations between via-points for each robot arm. These intermediate configurations include the joint positions, velocities and accelerations and must be supplied for each single control cycle (typically 16 milliseconds for most industrial robots). In producing this motion, different pre-requisites imposed by the operator and/or the manipulator design have to be tackled (e.g. arm physical constraints, minimum-time motion, minimum-energy motion [13]). In the literature straight-line, cartesian motion planning is usually employed, since it is claimed to have certain advantages in terms of motion correctness and ease of collision avoidance [10,14]. However, in an integrated manufacturing environment, a minimum-time motion (MTM) proves more attractive since fast completion of the tasks leads to higher productivity. Achieving such fast

motion requires planning in the joint space where time scaling of motion parameters is possible [15], and is more appropriate when producing the control commands [16]. Thus, time scaling of motion will ensure high-performance of the manipulator. However, collision avoidance must also be integrated in the procedure.

2.3. Collision Avoidance

When two (or more) manipulators operate in the same workcell, with no coordination the probability of a collision occurring is high. In addition, the fact that collisions may occur even if one of the arms is at rest adds a further complication to the planning necessary for single arm systems. Therefore, the planning procedure described above must be a feasible one in terms of producing a collision free motion to execute the required task.

Several methods for collision avoidance during robot motion are reported in the literature, including defining a free zone between the robots [2], representing all objects in the work space as polyhedrons [17], and placing a set of spheres at appropriate centre points along the manipulator limbs [10,14]. However, if a true cooperating robot system is sought, the first method is rendered as inefficient, since it effectively keeps the arms apart. In addition, the second method is expensive both computationally and in terms of storage. The third method involves comparing the distances between sphere centres on both arms which simplifies the calculations. In addition there is the possibility of varying the number of spheres used as needed by the application accuracy, as will be discussed in section 3.2.

2.4. Motion Tracking

Referring to figure (1), motion tracking is the final module before downloading the required motion to the robots, and it produces the control commands required by all joint motors employing a certain control law. This control law can be as simple as a PID compensator using the error between the given (desired) motion parameters and the actual ones provided by the feedback loop, or as sophisticated as a model-reference adaptive controller [18] employing the complicated and computationally expensive dynamic equations of motion [19] to produce the torque values.

However, when undergoing multi-arm cooperating actions, a stable and accurate control scheme is needed. This is particularly important when handling delicate and/or dangerous materials in hazardous environments. In addition, since fast motion is preferred to increase productivity, the fact that such motion can only be achieved efficiently using switching-points and bang-bang control [20] puts a further constraint on the control law to compensate for any motion deviations caused by vibrations or structural resonance of the manipulator material.

3. Motion Coordination of Two Robots

A typical multi-arm system is shown in figure (2), where a 4-robot workcell is illustrated. A *local* coordinate system is located for each arm, but once all arms are operating together motion must be represented with respect to a *global* coordinate frame within which all cartesian positions and kinematic transformations can be unified.

At this phase of the system design only motion coordination is considered. The crucial task is to have all arms operating in the unified work space but with no actual interaction between the various grippers (i.e. no closed-chain assemblies). This strategy is considered very appropriate since the wrist assembly usually has to reach its designated destination before actually performing the required cooperative task. Two algorithms, for minimum-time motion planning and collision avoidance, are developed for this purpose.

The motion planner employed here uses the point-look-ahead concept, where the via-points necessary for the construction of the trajectory are not chosen by the user a-priori to planning, but rather selected as the most suitable points by on-board sensors whenever needed. Once a look-ahead point is provided, planning commences in real-time followed by tracking the produced motion, and the process is repeated for each additional look-ahead point detected. Hence, while the manipulator hand is traversing one present segment, another next segment is being computed by the planner, based on the sensors advice. This approach is vital to achieve real-time operation and relaxes the constraint on having a structured environment.

3.1. Minimum-Time Motion Planning

In order to achieve high performance of the workcell, minimum-time motion (MTM) must be implemented, where the velocity of each of the arm joints during a specific motion is set to the maximum value possible. The continuity of both the position and velocity profiles is essential for executing the motion. However, although discontinuities in the acceleration profile may cause vibrations during motion, it does not cause a disruption of execution. In developing the MTM algorithm, two *via-points* are supplied by the task planner specifying the *start* and *end* points in joint space, for which the continuity of motion must be respected. The procedure described hereafter applies for each of the arms independently.

3.1.1. *The Planner Formulation*

For each set of end-points supplied for a joint motion, a number (n) of intermediate via-points is computed as a linear variation between the two points, thus providing the necessary number of points to perform a cubic spline [21]. Since maintaining high velocity movement leads to a reduction in the travelling time, extreme velocity points are located on the splines constructed and set to a maximum. Furthermore, to achieve minimum travelling time between these maximum values of velocity, linear variation is imposed thus giving a quadratic equation for position. The scaling of the motion performance is achieved using a factor K defined as [22]

$$K = \max \left[\max_t \frac{\dot{\theta}(t)}{\dot{\theta}_{limit}} , \sqrt{\max_t \frac{\ddot{\theta}(t)}{\ddot{\theta}_{limit}}} \right] \qquad (1)$$

where θ, $\dot{\theta}$ and $\ddot{\theta}$ denote joint positions, velocities and accelerations, and θ_{limit} is the physical limit of the parameter. The new values of the motion parameters and time interval h are given as

$$\dot{\theta}_{new} = K^{-1}\,\dot{\theta}\,, \quad \ddot{\theta}_{new} = K^{-2}\,\ddot{\theta}\,, \quad h_{new} = K\,h \tag{2}$$

The above augmentation of cubic and quadratic position profiles produces a discontinuous acceleration profile in the form of *bang-cruise-bang*, which is reported to be the best efficient minimum-time motion [23]. Hence, the joint will transit smoothly in the vicinity of an imposed via-point, but otherwise uses a bang-bang motion where sudden switching of acceleration limits is tolerated [16].

3.1.2. *Grid Search*

In reality, there is an infinite number of possible motions to move from one point to another in space [20]. Therefore, the minimum-time motion produced is not necessarily the best choice, and the work space must be searched for possible more-optimum options [24] by employing a proper grid search to cut down on the computational complexity involved. This grid search is performed by varying the original imposed via-points (except the end-points) and the new motion produced is checked as a possible better choice. Hence, a cost value T_{cost} is defined for each interval h between two successive via-points as

$$T^{i}_{cost} = h^{i}_{old} - h^{i}_{new}\,, \qquad i=1,2,...,n+1 \tag{3}$$

and a maximum for T^{i}_{cost} is sought. In this way, local segments from different options of motion are collected to produce the best optimum choice [25, pp. 30-66]. The general structure of the above procedure is illustrated in figure (3) for a four via-point motion, where the chosen optimum is taken amongst different options as shown in bold in the diagram.

In addition to producing the best optimum, the MTM algorithm saves all other possible options as alternatives. These will be used once additional constraint of collision avoidance is considered.

3.2. Collision-Free Motion Checking

Although a feasible and efficient motion has been planned for all the arms as described in the last section, the fact that they all share a single work space requires ensuring a collision-free (CF) performance. Hence, each point on all motion trajectories must be checked and altered if a collision is detected. Consequently, other less-optimum options saved by the MTM algorithm may be considered if the best choice is not feasible.

3.2.1. *Assigning the spheres*

As discussed in section 2.3, the method of setting spheres on the manipulator links is efficient and this will be used to implement the CF algorithm. The CF can only be activated once the MTMs for all arms are terminated. For simplicity, and also for coherence with the practical implementation of section 4, the CF will be illustrated for collision detection between two robots only, although it can be readily generalised.

For each of the manipulators, each link is decomposed into a number of spheres, the origins of which are placed a certain distance apart. The radius of each of the spheres can be determined by [26]

$$r_i = \sqrt{\left(\frac{l_i}{2M_i} \right)^2 + \left(\frac{w_i}{2} \right)^2}, \quad i=1,...,L \tag{4}$$

where l_i and w_i denotes the length and width of the particular link and L denotes the total number of links on the arm. The number M_i is the total number of spheres used, and can be set by the user according to the application. However, precautions must be taken when the manipulator has large obstructive motor or gear assemblies which are better modelled collectively as one sphere. The spheres are allocated along the length of each link and the position of their centres calculated as

$$C_i = \frac{(2m_i + 1) (p_i^s - p_i^e)}{2M_i}, \quad m_i=1,...,M_i \tag{5}$$

where p^s_i and p^e_i denote positions for the start point and end point of link i. Such a decomposition can be generalised to any number of links L, for which the radius r may be varied depending on the shape of the manipulator. To illustrate the sphere allocation procedure a spherical model of two 3-link articulated arms is shown in figure (4a).

The number of spheres used in the collision detection algorithm is chosen according to the application and as specified by the operator. Hence, the number of spheres can be chosen to cover the link leaving a certain clearance as desired, where the larger the number of spheres the less the clearance around the link, as illustrated in figure (4b).

3.2.2. *Checking for collisions*

Once the spheres are allocated, the collision avoidance procedure can be activated. The distance between each centre of a sphere on one arm and each centre of sphere on the other arm is calculated and compared against the sum of the relevant sphere radii. Thus, a cost value is defined as

$$d_k < r_i^1 + r_j^2, \quad i=1,..,M_s^1, \quad j=1,..,M_s^2 \tag{6}$$

where d_k denote the distance between the two centres, r^1_i and r^2_j are the radii of spheres on arms one and two respectively and M^a_s, a=1,2 is the total number of spheres on both arms. If equation (6) is valid, then no collision occurs, otherwise the two arms collide for this particular point, as illustrated in figure (4b). The above comparison procedure is repeated for each point on the robots' motion, i.e. the CF is activated within each control cycle. In addition, since the MTM planning procedure was performed in the joint space while the CF algorithm operates in the Cartesian space, the direct kinematic equations must be computed for each of the sphere centres used by equation (6), and again within each control cycle.

The above CF procedure is initially given the data corresponding to the two optimal motions produced by the MTM for both arms. However, if a collision is detected during this motion, other possible less-optimum options are considered to obtain a feasible solution. Thus, each of the near-optimal options of a robot is checked, in turn, with other possible options of the other robot. This approach will reduce the possible combinations of options and thus lower the possibility of obtaining a solution. However, due to the computational complexities associated with the CF mentioned above, such a compromise is acceptable. If required by the user, further motion combinations can be included in the algorithm.

Refering to figure (4a), it can be seen that although the wrist positions of both robots are well apart, the lower links of the arms collide as indicated by the overlapping of spheres on link 2 of both robots. This will be illustrated further in the results reported in section 5.

3.2.3. *Finding a Feasible Choice*

It is possible that after examining all possible options, a feasible solution cannot be found to complete the required task. In this case the part of the motion that is feasible is executed and an overall new process including both the MTM and CF algorithms is initiated. In this new procedure, the start points are set to the final positions reached by the previous unsuccessful motion, while the end points remain as before, since these are the goals. This continuing procedure is continued until the task is completed, as illustrated in table (1).

4. Implementation of the Workcell

This section discusses practical implementation of the multi-arm system (MAS) as used above. The MAS consists of two, 6 d.o.f. RTX robot manipulators controlled by a Sun SPARC-IPC workstation. Referring to figure (2), the two left-hand-side robots are designated as RTX #1 and RTX #2, and the separating distance is D=1.05 metres. The general hardware setup of the system is shown in figure (5), where the interface boards convey the control commands to the robot motors as well as feeding back the encoder counts to the controller [27]. All the system control modules are written in C, including the CF-MTM algorithm and the PID compensator. The following arrangements are made for the practical implementation:

- Due to the lack of intelligent capabilities and advanced sensory equipment on the system at this stage of development (e.g. advanced vision systems [28]), the required via-points are simply read from a datafile where it is stored using a teach-by-hand method.

- For the MTM algorithm, the RTX physical constraints are taken into consideration [29] to ensure a realistic motion. Due to the memory limitations, the total number of motion options was restricted to ten per joint. Hence, although the search technique employed in the planning algorithm is a global one and investigates all possible options within the physical limits of the joint position, considering the requirement for fast computations in a real-time application and the limitations on the memory available, the number of search passes must be limited, thus limiting the system performance. Nonetheless, the results presented in section 5 are adequate to demonstrate the practical implementation of the system.

1. Task planner: provide start and end points for both robots

2. For each robot Do

    Execute MTM:

        Find optimal option

        Save other less-optimum options

3. Execute CF using optimal options of both robots

4. While CF motion is not satisfied DO

    For all other options of robot #2 DO

        Execute CF using optimal option of robot #1 and one option of robot #2

    For all other options of robot #1 DO

        Execute CF using optimal option of robot #2 and one option of robot #1

5. If CF not satisfied : Reset start points

6. Execute CF-MTM again; (goto step 1)

Table (1): The CF-MTM Algorithm

■ Considering the CF algorithm, a total of ten spheres were used on each of the first three links of the RTX arm. However, the spherical-type wrist assembly [30] is considered as one unit and assigned one single sphere. Hence a total of 31 spheres are set on each arm. This number can be increased for closer coverage of the manipulator if required (see figure 4b). In addition, to provide the positions of the centres of spheres in cartesian coordinates, the direct kinematic formulation of the RTX is used [31] to transform the planned joint positions. These equations are simplified by the fact that the robot wrist is frozen.

■ The control law, programmed and run as a separate module on the SPARC, is a simple PID compensator similar to that on the original system [29]. This is quite appropriate at this stage of the system development since performance comparisons will be made between the original system and the designed one, as will be reported in section 5.

These arrangements are chosen to facilitate the action of the prototype system and

would be waived in more comprehensive system versions, where more computational power may be available. The primary benefit remains in demonstrating the applicability and feasibility of the developed algorithms in a practical real-time system. An overall view of the MAS is shown in figure (6).

5. Case Study Results

In testing the MAS implementation, the MTM algorithm was executed for all six joints of each of the RTX arms. Due to memory limitations, however, the total number of search options was limited to 10, including the optimal choice. Since planning is performed in the joint space, this assumption may limit the performance of the algorithm. For the CF algorithm, a unified sphere radius was chosen for each limb of the arm and computed from equation (4) with r_b = 0.090 m (for the base), r_{se} = 0.061 m (for both the shoulder and elbow links) and r_w = 0.158 m (for the wrist assembly). Although the MTM produces the joint motions for the final three joints of the arm, using a frozen wrist while executing the CF was tolerated at this stage of the system development. In addition, the positions of the spheres were computed according to equation (5). Although other results could be included, the case study following is a typical one showing the capabilities of the CF-MTM algorithm to plan a feasible motion for the MAS to execute.

For this case study, two sets of start and end positions for the two RTX arms were set by a teach-by-hand method. The two sets of joint positions shown in table (2) were chosen to impose a collision during the execution of the original motion. Hence, the CF-MTM algorithm was required to produce a feasible motion to achieve the specified goals.

The 3-D graph in figure (7) illustrates the MAS workcell during the original motion. In figure (7), the base of the workcell is a square of side D=105 cm, with the origins of RTX #1 and RTX #2 at cartesian positions (0.0,0.0,-1.25) metres and (1.05,0.0,-1.25) metres, respectively. To show the motion of both arms clearly, only the positions of the first three joints are plotted, giving the time history of both the elbow and wrist motions in addition to a single point showing the shoulder position. The motion of the first RTX is shown on the left-hand-side of the workcell, while the motion of the second RTX is on the right-hand-side.

Robot Manipulator		Joint Positions (J_1 in mm and all others in degrees)					
		J_1	J_2	J_3	J_4	J_5	J_6
RTX 2	Start	-130.95	69.19	-66.87	-79.88	-38.63	8.01
	End	-401.12	-59.99	-27.79	-55.75	-2.48	-3.11
RTX 1	Start	-130.95	69.19	-66.87	-79.88	-38.63	8.01
	End	-401.12	-59.99	-27.79	-55.75	-2.48	-3.11

Table (2): Via-points for case study motion

To give a better indication of the collision during the above motion, a top-view of the workcell is shown in figure (8), where the inner two curves show the wrist motions of both arms while P_s^i and P_e^i, i=1,2 denoting the start and end points of each. In addition, points A and B show the position where both wrists collided.

Applying the MTM produced a minimum-time motion executable by both the RTX controllers in 4.97 seconds (311 control cycles at 16 milliseconds each), as compared to about 10 seconds for the original motion planned and executed by the original RTX system, hence the term *original* in figures (7) and (8). Other less-optimal options, a total of 8 in this example, are shown in table (3) for both manipulators. However, once the CF motion was checked, the original optimal choice was found to drive the robots into a collision after executing 99 control cycles (1.58 seconds) as shown in figure (9), where the distance between the spheres mounted on each wrist was detected between points A and B as 0.314 m which is less than $2r_w$ = 0.316 m, and the other stored options had to be checked in turn. The optimal motion of robot 2, together with the second option of robot 1, were found to give a feasible collision-free motion. Although the second option of robot 1 required 8.66 seconds to execute (541 control cycles) as compared to only 4.97 seconds for the optimal option, it was the *best possible* choice and had to be accepted.

Robot Arm	Possible Motion Options (seconds)							
	1	2	3	4	5	6	7	8
RTX #1	4.97	8.66	8.87	8.97	9.09	11.30	11.34	12.64
RTX #2	4.97	8.66	8.87	8.88	8.94	10.92	11.03	11.30

Table (3): Possible MTM Options

In conclusion, when both robot motions were executed, RTX #2 reached its goal in 4.97 seconds while RTX #1 needed an extra 3.69 seconds to reach the end point due to the particular motion options selected. A top view of the workcell executing this chosen CF-MTM motion is shown in figure (10), where point C shows the position of the first RTX when the

second RTX completed its motion and was at rest at P_e^2, while RTX #1 continued from C ro P_e^1. This describes the reality of the motion, although the plot of figure (10) gives the impression that a collision occurs. With the lack of possiblity to include a time-scale in figure (10) it must be emphasised that while the wrist of RTX #1 is still at point C, the wrist of RTX #2 is already at P_e^2, thus eliminating the possibility of collision.

The assessment of the execution time of the CF-MTM algorithm varies depending on the search passes (automatically determined by the MTM) and the number of spheres chosen for the CF. In any case, the planning for a motion segment is done while executing the previous motion segment and is therefore relative to the execution time of the motion segment. For this particular case study, the execution time is ..... minutes. However, measures to keep the execution time in a lower value may be taken (e.g. limiting the number of spheres and search passes) as described in section 4.

6. Further Work

The control engineering research group at QUB is involved with the design of an integrated multi-manipulator workcell [32]. Previous research activities concentrated on the design of a transputer-based controller for the RTX robot, with all the required hardware and software interfaces [33,34].

In implementing the MAS, achieving collision-free minimum-time motion for two coordinated multi-joint robot arms involves heavy computational burdens and forces certain compromises in executing the procedure as discussed in section 4. Thus, realising the need to implement the system for a fast real-time industrial application, recent on-going research is concerned with the distribution of different control modules on a multi-processor environment, thus providing more powerful computing abilities to accommodate for all the MAS requirements. This approach is encouraged by previous successful implementations of parallel robot structures on transputers [35,36].

Nonetheless, the primary benefit in presenting the MAS remains in providing the appropriate algorithms in a practical implementation of a rather complicated robot control system.

One further stage in the system development is to include 3-D graphics for the on-line presentation of the arms movement within the workcell, thus providing an excellent man/machine interface for the monitoring and analysis of the integrated environment [37] (see figure (1)).

7. Conclusions

The MAS is a working multi-arm environment where the coordinated motion of two RTX arms is planned and executed. The high-performance of the system demonstrated by executing minimum-time movements along with the ability to avoid collisions is of significant importance in an automated workcell. In implementing the MAS, the real problems associated with the complicated multiple robot control are tackled, especially when a decision must be

made on choosing the feasible motion. The case study results reported in section 5 demonstrates the ability of the system to detect and successfully avoid collisions while planning the execution of motion. The system so far is considered as a development platform, where other arms can be added (see figure (6)), in addition to investigating other aspects of multi-arm control in due course [38].

Acknowledgement

The group acknowledges the help and support of the Technology Board for Northern Ireland (Grant no. ST79) and the SERC/DTI transputer Initiative.

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] Roach, J.W. and Boaz, M.N., 'Coordinating the motions of robot arms in a common workspace', *IEEE J. Robotics and Automation*, Vol. 3, No. 5, pp. 437-44, 1987.

[3] Tao, J.M., Luh, J.Y.S. and Zheng, Y.F., "Compliant Coordination Control of Two Moving Industrial Robots", *IEEE Trans. Robotics and Automation*, Vol. 6, No. 3, pp. 322-30, 1990.

[4] Zheng, Y.F., Luh, J.Y.S., "Optimal load distribution for two industrial robots handling a single object", In Proc. IEEE Int. Conf. Robotics and Automation, pp. 344-9, 1988.

[5] Li, Z., Tarn, T.J. and Bejczy, A.K., "Dynamic workspace analysis of multiple cooperating robot arms", *IEEE Trans. Robotics and Automation*, Vol. 7, No. 5, pp. 589-96, 1991.

[6] Tarn, T.J., Bejczy, A.K. and Yun, X., "Design of dynamic control of two cooperating robot arms: closed chain formulation", In Proc. IEEE Int. Conf. Robotics and Automation, pp. 7-13, 1987.

[7] Cheng, F.-T. and Orin, D.E., "Efficient formulation of the force distribution equations for simple closed-chain robotic mechanisms", *IEEE Trans. Sys., Man and Cyber.*, Vol. 21, No. 1, pp. 25-32, 1991.

[8] Murphy, S.H., Wen, J. T.-Y. and Saridis, G.N., "Simulation of cooperating robot manipulators on a mobile platform", *IEEE Trans. Robotics and Automation*, Vol. 7, No. 4, pp. 468-77, 1991.

[9] Freund, E., "On the design of multi-robot systems", In Proc. IEEE Int. Conf. Robotics and Automation, pp. 477-90, 1984.

[10] Lee, B.H. and Lee, C.S.G., 'Collision-free motion planning of two robots', *IEEE Trans. Sys. Man and Cyber.*, Vol. 17, No. 1, pp. 21-32, 1987.

[11] Nagata, T., Honda, K. and Teramoto, Y., "Multirobot plan generation in a continuous domain: planning by use of plan graph and avoiding collisions among robots", *IEEE Trans. Robotics and Automation*, Vol. 4, No. 1, pp. 2-13, 1988.

[12] Cui, X. and Shin, K.G., "Intelligent coordination of multiple systems with neural networks", *IEEE Trans. Sys., Man and Cyber.*, Vol. 21, No. 6, 1488-97, 1991.

[13] Vukobratovic, M. and Kircanski, M., "A method for optimal synthesis of manipulation robot trajectories", *Trans. AMSE, J. of Dyn. Syss., Meas. and Control*, Vol. 104, pp. 188-93, 1982.

[14] Basta, R.A., Mehrota, R. and Varanasi, M.R., 'Detection and avoiding collisions between two robot arms in a common workspace', In *Robot Control: Theory and Applications*, pp. 185-92, 1988.

[15] Hollerbach, J.M., 'Dynamic scaling of manipulator trajectories', *Trans. ASME, J. of Dyn. Syst., Meas. and Control*, Vol. 106, pp. 102-6, 1984.

[16] 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.

[17] Lozano-Perez, T., 'Automatic planning of manipulator transfer movements', *IEE Trans. Syst., Man and Cyber.*, Vol. 11, pp. 681-98, 1981.

[18] Fu, K.S., Gonzalez, R.C. and Lee, C.S.G., *Robotics: Control, Sensing, Vision and Intelligence*, McGraw Hill, New York, 1987.

[19] Hollerbach, J.M., 'A recursive lagrangian formulation of manipulator dynamics and a comparative study

of dynamics formulation complexity', *IEEE Trans. Syst., Man and Cyber.*, Vol. 10, pp. 730-6, 1980.

[20] Bobrow, J.E., Dubowsky, S. and Gibson, J.S., 'Time-optimal control of robotic manipulators along specified paths', *Int. J. Robotics Research*, Vol. 3, No. 3, pp. 3-17, 1985.

[21] Hildebrand, F.B., *Introduction to Numerical Analysis*, McGraw Hill, 1956.

[22] Zalzala, A.M.S. and Morris, A.S., 'Structured motion planning in the local configuration space', *Robotica*, Vol. 9, No. 1, pp. 81-92, 1991.

[23] Kim, B.K. and Shin, K.G., 'Minimum-time path planning for robot arms and their dynamics', *IEEE Trans. Sys., Man and Cyber.*, Vol. 15, pp. 213-23, 1985.

[24] Sahar, G., and Hollerbach, J.M., 'Planning of minimum-time trajectories for robot arms', *Int. J. Robotics Research*, Vol. 5, No. 3, pp. 90-100, 1986.

[25] Zalzala, A.M.S., 'Fast minimum-time trajectory planning and control of intelligent robots with VLSI implementation', *Ph.D. Thesis*, Department of Automatic Control and Systems Engineering, University of Sheffield, UK, 1990.

[26] Beaumont, R.G. and Crowder, R.M., 'Two-armed robot systems - A review of current theory and development of algorithms for real-time collision avoidance',

[27] Wilson, G., 'A transputer-based control system for the RTX robot', *Internal Report*, Dept. of Electrical and Electronic Engg., The Queen's University of Belfast, 1991.

[28] Porrill, J., Pollard, S.B., Pridmore, T.P., Bowen, J.B., Mayhew, J.E.W. and Frisby, J.P., 'TINA: the Sheffield AIVRU vision system', *AIVRO memo #27*, AI Vision Research Unit, Sheffield University, UK, 1988.

[29] Universal Machine Intelligence Ltd., *Inside RTX*, London, 1986.

[30] Paul, R.P. and Zhang, H., "Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation", *Int. J. Robotics Research*, Vol. 5, No. 2, pp. 32-44, 1986.

[31] Song, Y. and DeKeyser, R.M.C., "Inverse kinematics of the RTX robot: Geometric approach and the solution of non-uniqueness", In Proc. American Control Conf., Vol. 2, pp. 1780-5, 1990.

[32] Dodds, G.I. and Irwin, G.W., 'Planning a collision-free cooperating robot environment', In Proc. IMA Conf. on Robotics, Loughborough, UK, 1989.

[33] Dodds G.I., 'Transputer hardware and software for a multimanipulator environment', In IEE Colloquium on Parallel Processing in Control - The Transputer and Other Architectures, Digest no. 1990/050, London, 1990.

[34] 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.

[35] 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.

[36] 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.

[37] Dodds, G.I. and Ogasawara, T., 'Transputer based simulation of a teleoperated manipulator', In Proc. 9th Annual Conf. of the Japanese Robotics Society, Tsukuba, Japan, 1991.

[38] Dodds, G.I., Irwin, G.W., Zalzala, A.M.S. and Ogasawara, T., "Basic architectures and processing for multi-robot control", In Colloquium on Applications of Parallel and Distributed Processing in Automation and Control, 1992.

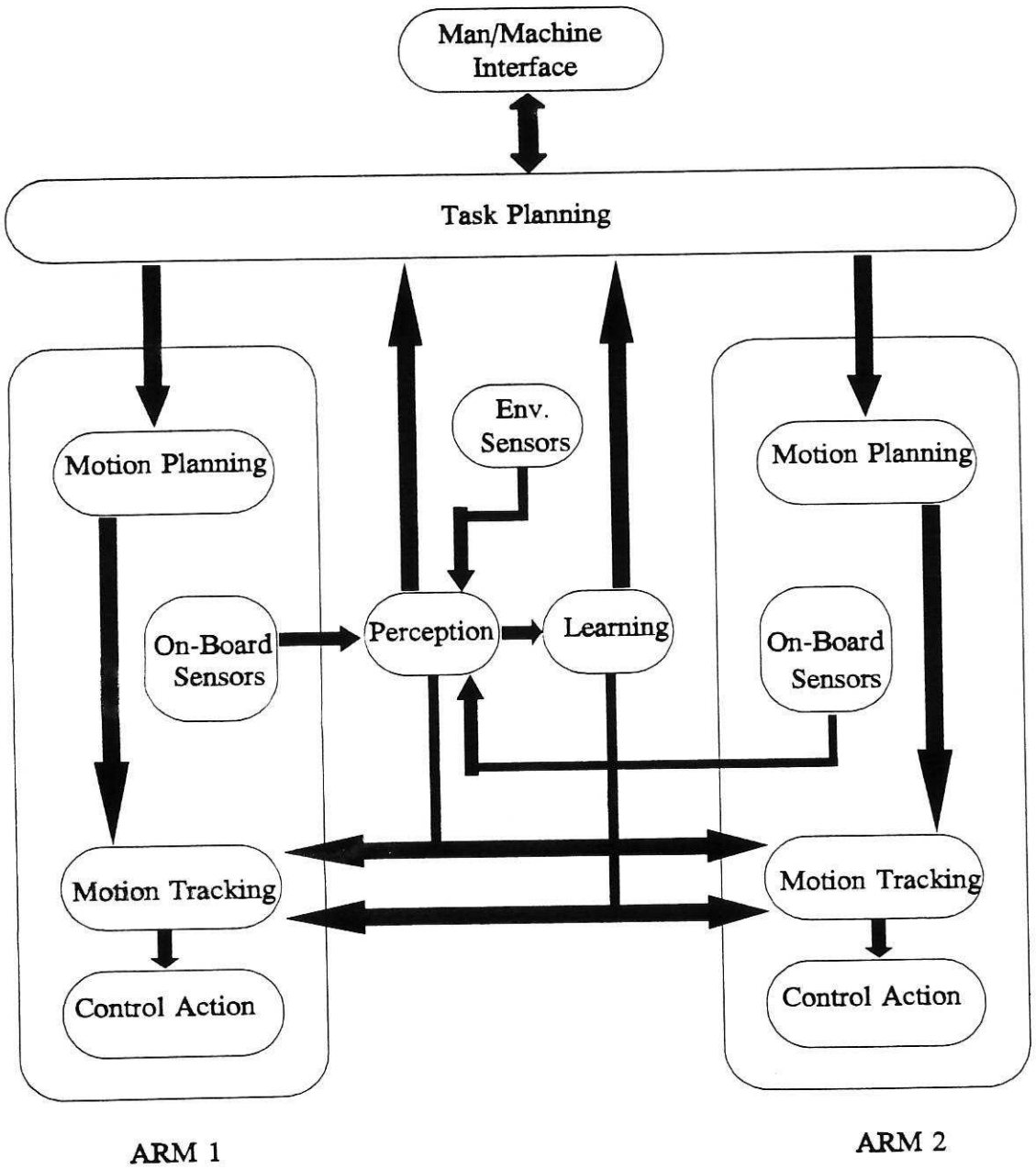


Figure (1): General Multi-Robot Control Structure

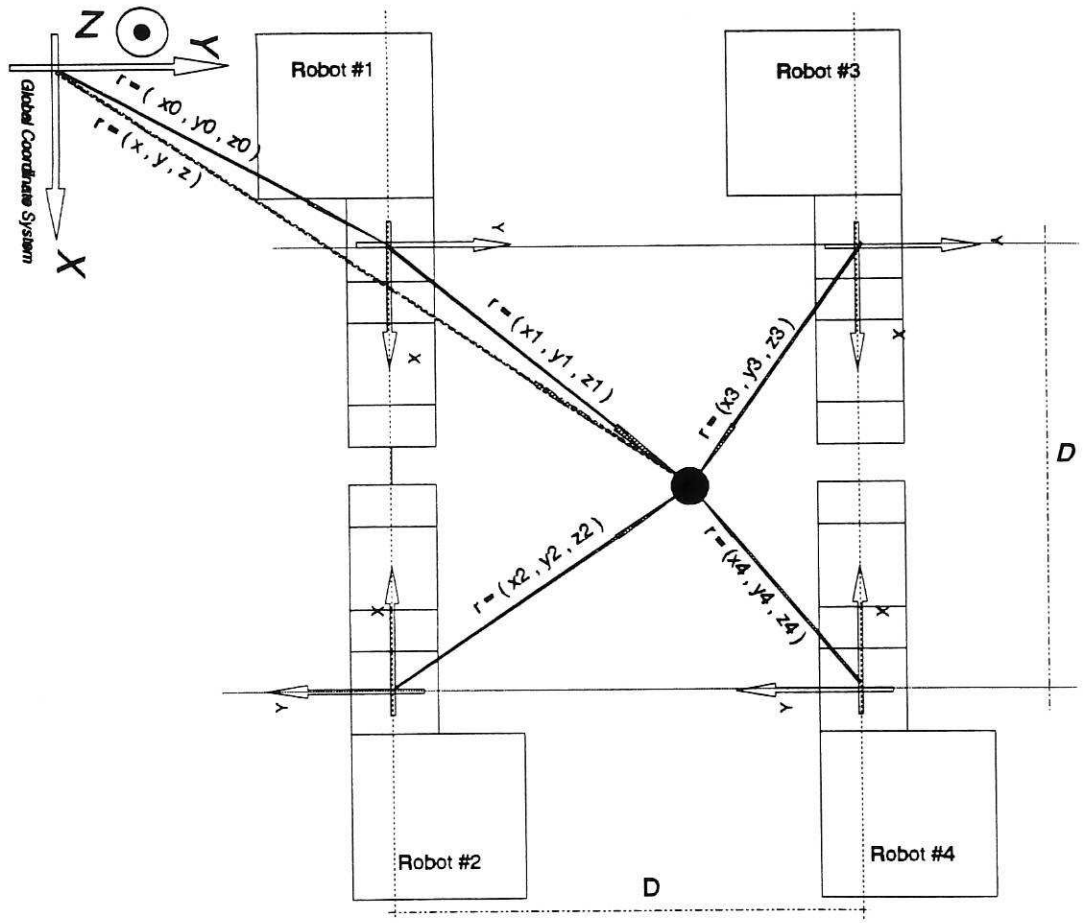


Figure (2): Typical Multi-Arm Environment

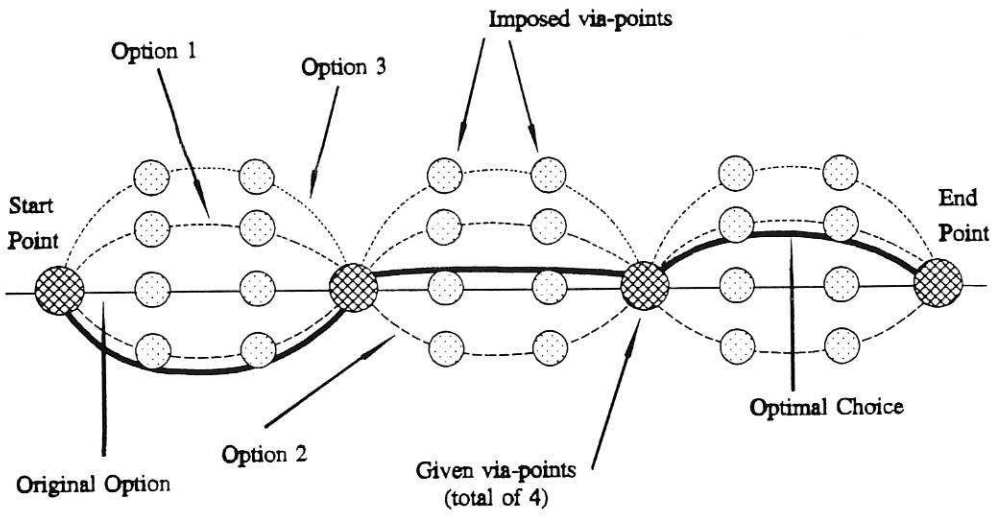
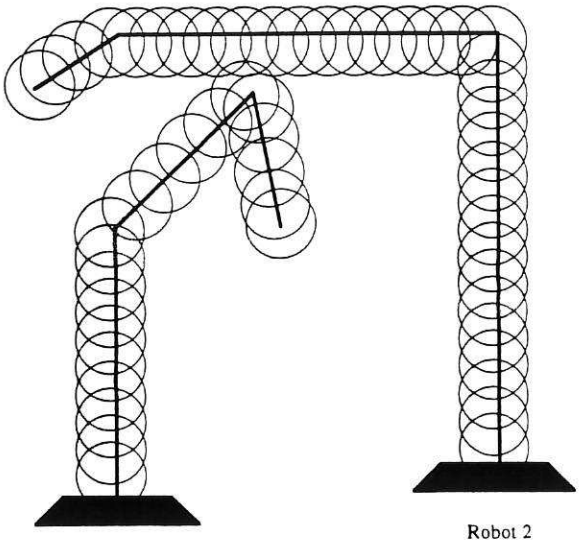
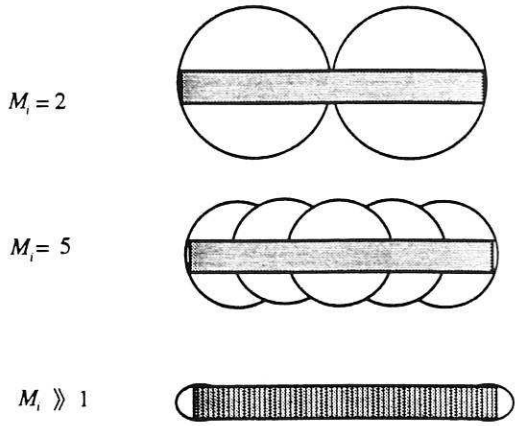


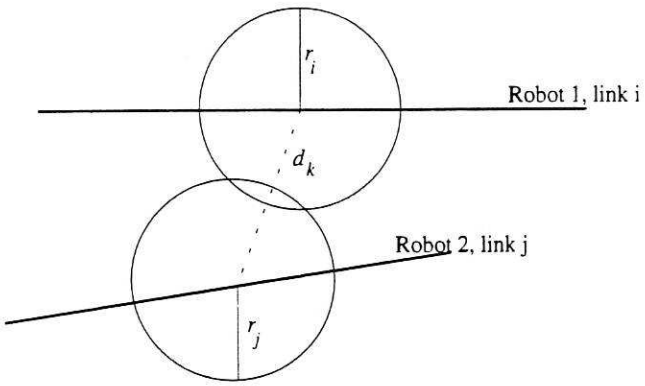
Figure (3): The MTM Procedure in Joint Space



a. Spheres covering the two arms



b. Choice of number of spheres on a link



c. Collision between two spheres

Figure (4): Assigning the Spheres

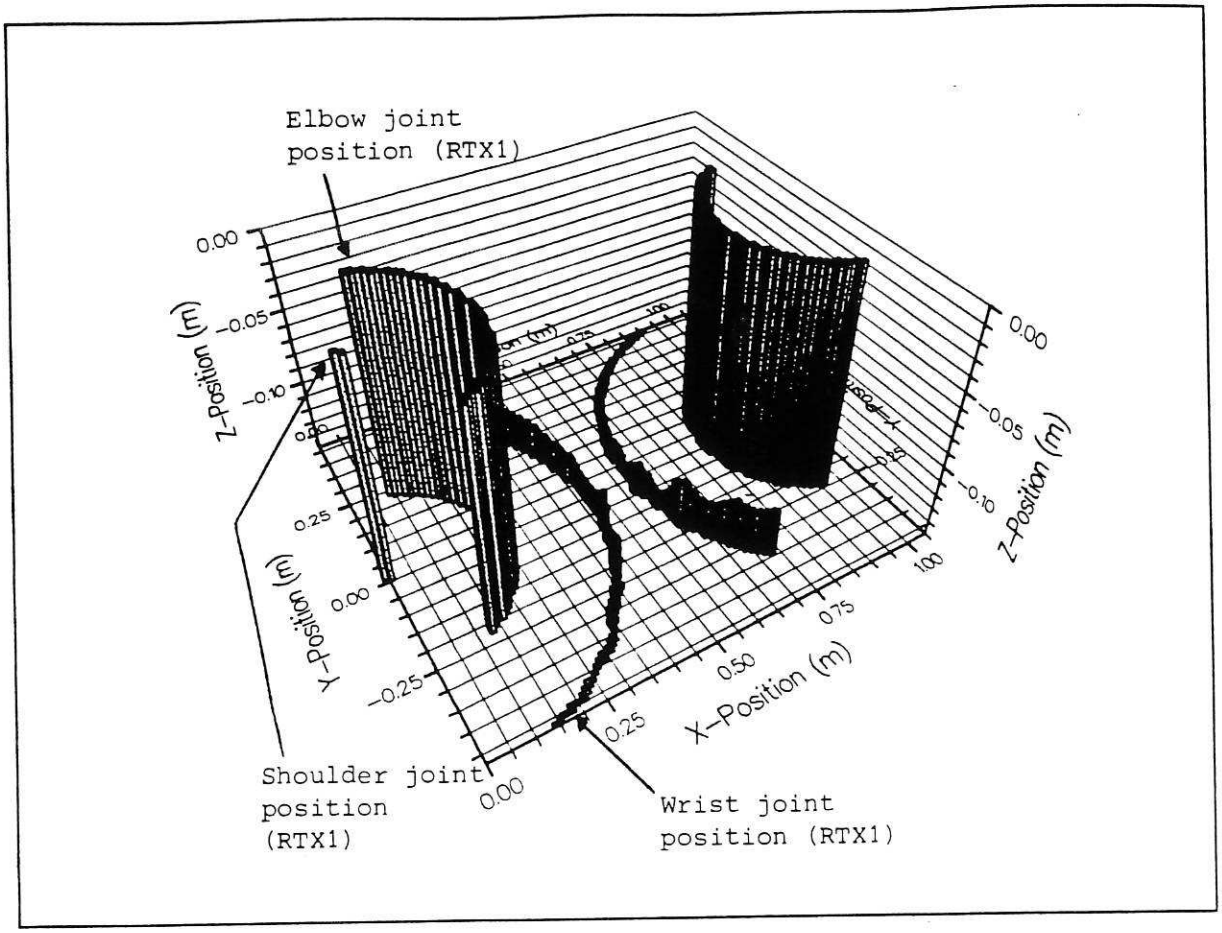


Figure (7): Graphical 3-D plot of the MAS workcell

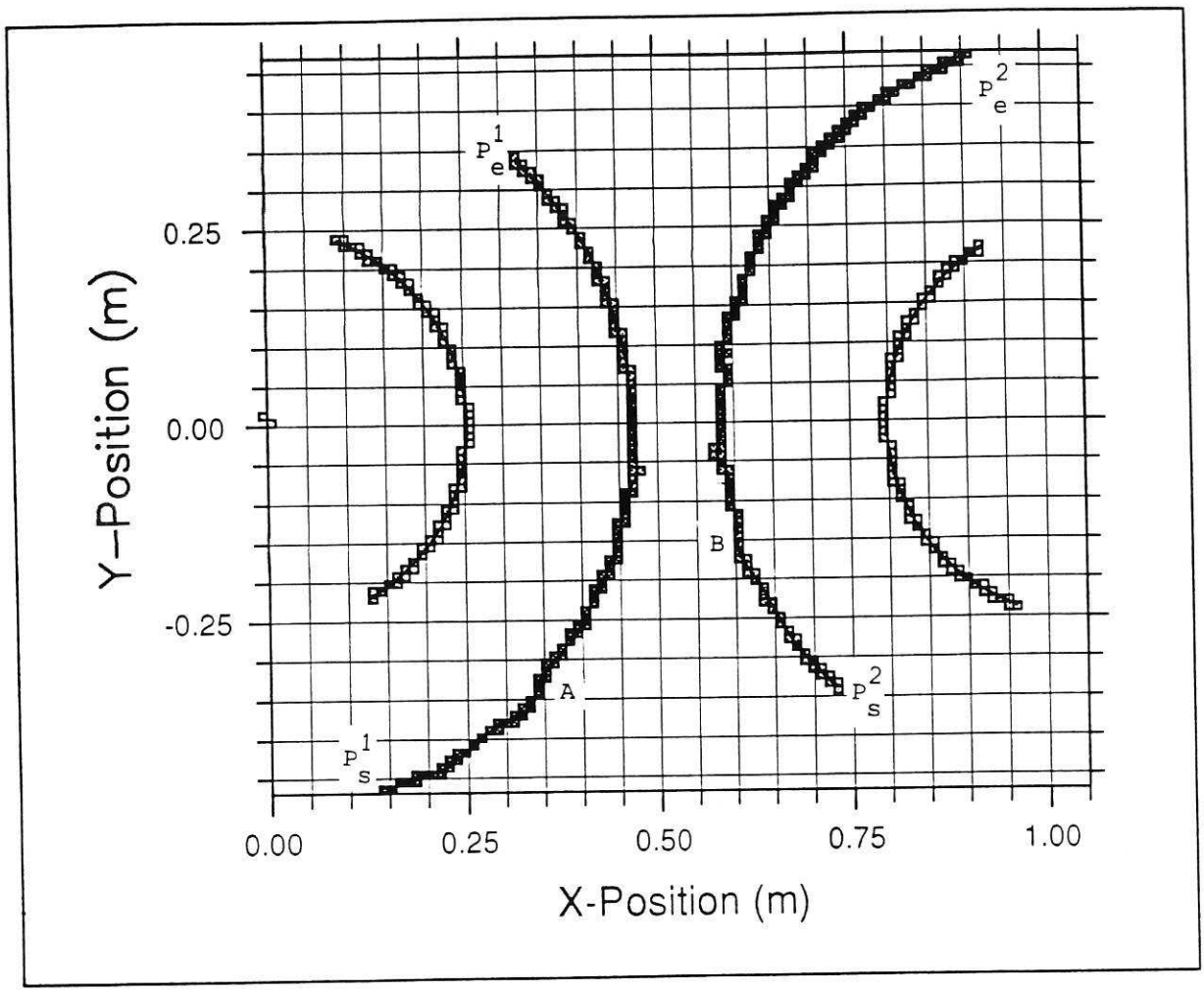


Figure (8): Top-view of the optimal planned motion

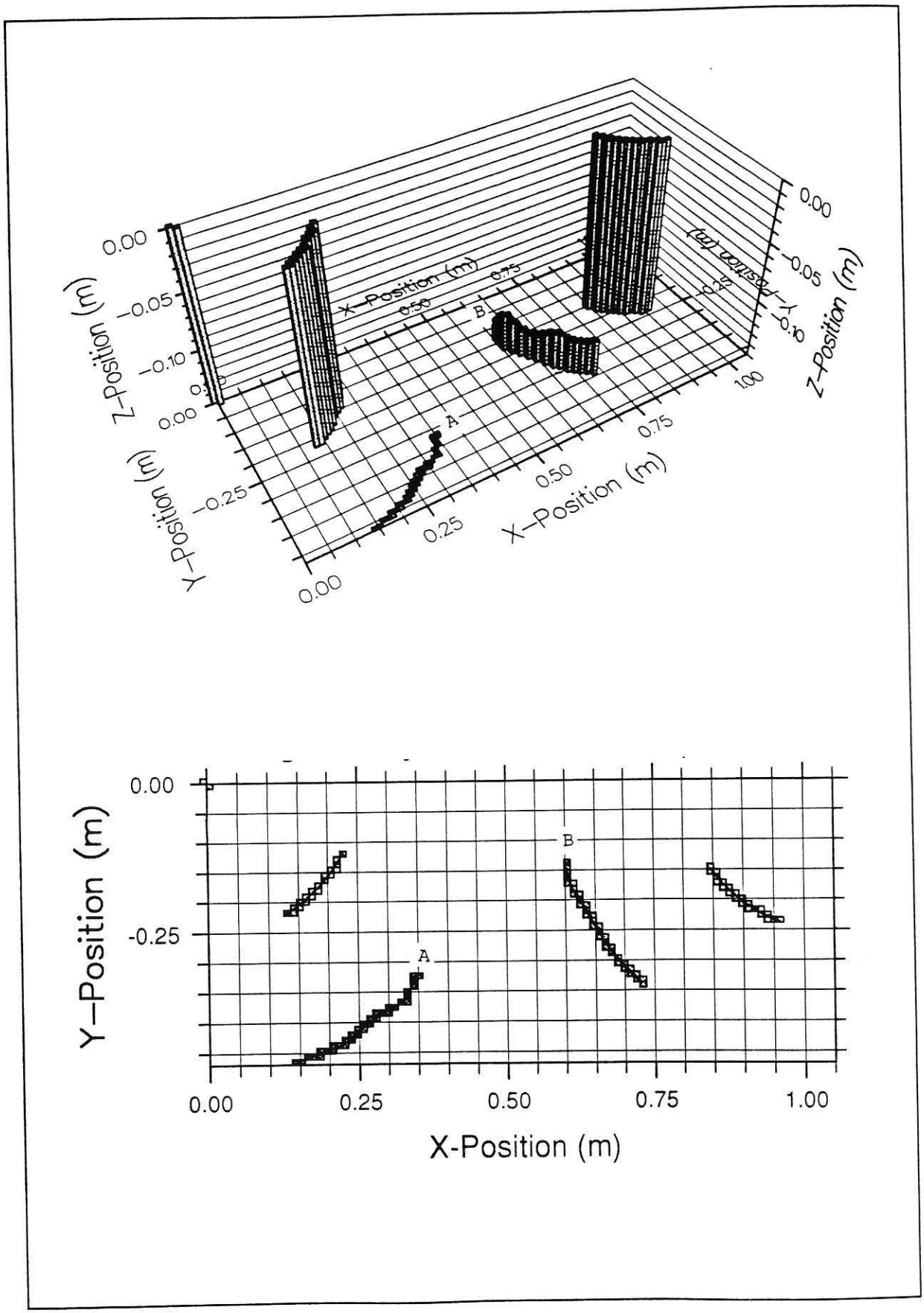


Figure (9): Point of collision during the optimal motion

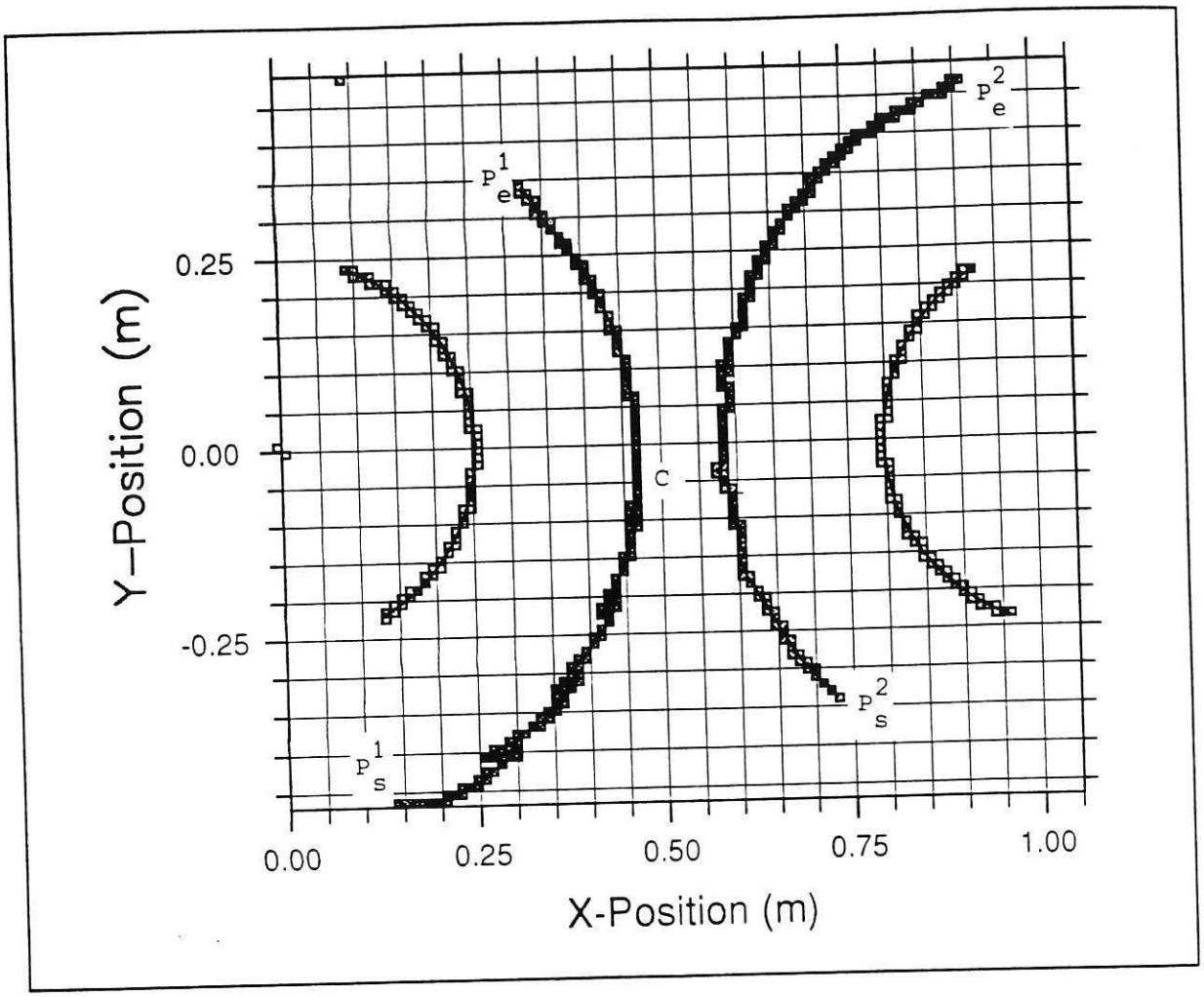


Figure (10): Top-view of the planned CF-MTM motion

SHEFFIELD UNIV. LIBRARY ST. GEORGE'S