ELSEVIER

Robotics and Autonomous Systems 12 (1994) 15–27

Robotics and Autonomous Systems

A cell mapping method for general optimum trajectory planning of mulitple robotic arms

Fei-Yue Wang *,[a], Paul J.A. Lever [b]

[a] *Systems and Industrial Engineering Department, The University of Arizona, Tucson, AZ 85721, USA*
[b] *Mining and Geological Engineering Department, The University of Arizona, Tucson, AZ 85721, USA*

Abstract

This paper proposes a method that uses cell state space and cell mapping based techniques for planning general optimum trajectories along given geometric paths for coordinated multiple robotic arm systems. The major advantages of this method include its simplicity and applicability to a wide range of problem formulations. In particular, three performance indices for optimum trajectory specification are investigated, i.e., minimum-energy, minimum-jerk, and minimum-time formulations. A simple search strategy is constructed using cell-to-cell mapping to find optimum trajectories. A special feature of this search algorithm is its ability to generate all optimum trajectories for all possible initial conditions through a single search. The computational complexity is analyzed for the search algorithm and its hierarchical implementation. Parallel execution of the hierarchical search method is discussed and the results indicate that it can improve the cell-mapping search efficiency significantly.

Key words: Cell mapping; Computational complexity; Energy-optimal trajectory; Hierarchical search; Jerk-optimal trajectory; Multiple arms; Time-optimal trajectory; Trajectory planning

1. Introduction

Optimum trajectory planning is of major importance for effective control of robotic arm systems. During the past decade considerable effort has been made to solve the problem for single robotic arms [1,14,17,19,13,10]. In particular, the generation of time-optimal trajectories is the focus for most of the research work [16,4,21,2,20], since the productivity of manufacturing systems is directly related to the execution time of robotic motion tasks.

However, only a few studies on the corresponding problem for multiple robotic arms have been reported [18,9,3,22]. The main reason is due to the problem of high nonlinearity and strong coupling in their dynamics. But coordinated multiple arm systems are essential for many applications. These include, robotic tasks in manufacturing assembly, mining transportation, and structure construction for space exploration. In addition, optimum trajectories other than time-optimal are also important for multiple

* Corresponding author.

SSDI 0921-8890(93)E0049-S

arm systems. For example, an energy-optimal trajectory is commonly desired in environments where a limited power supply is available (i.e., robotic motions for Lunar or Martian missions [23]) or where repetitive robotic operations are required over long time periods, while a jerk-optimal trajectory is useful for manipulation tasks where high precision and smooth motion must be maintained [11]. These optimum trajectory planning problems have not yet been well studied in the robotics literature. A general technique to solve these problems must be developed in order to ensure the optimal operation of multiple robotic arm systems in those situations.

The cell mapping method is considered to be a simple and efficient solution to optimum control [8,24] and trajectory planning problems [21,22]. In our previous work [22], this method has been used to solve the problem of time-optimal trajectory generation for a coordinated multiple robotic arm system handling a common object along a prescribed geometric path. The dynamic equations for coordinated arms were transformed into a set of second-order differential equations with a single path parameter. Force/torque constraints imposed on the arm joint actuators were expressed in terms of upper and lower bounds on the path parameter acceleration using a modified linear programming method. This led the original time-optimal trajectory generation problem to an optimal control problem in a two-dimensional phase plane. A cell-to-cell mapping was constructed from the continuous state equation by assigning a cellular structure to the phase plane. Using the special structure of time-optimal control and cell-to-cell mapping a simple search algorithm was applied to find the time-optimal trajectory. A hierarchical implementation of the search algorithm that allows for parallel execution was also discussed. The primary goal of the current paper is to generalize this cell-mapping based technique for optimum trajectory planning of coordinated multiple robotic arms with general performance index specifications.

This paper is divided into six sections. First, dynamic equations for coordinated multiple robotic arms moving a common object along a given geometric path are discussed in Section 2. Section 3 formulates optimum planning problems for obtaining minimum-time, minimum-energy, and minimum-jerk trajectories. The basic concept of the cell mapping method and its application to optimum trajectory planning is described in Section 4, where a simple trajectory search strategy is also constructed using cell-to-cell mapping. Section 5 presents an analysis of computational complexity for the search algorithm and its hierarchical implementation. The efficiency of the hierarchical search method is demonstrated. Finally, some concluding remarks and suggestions for future work are given in Section 6.

2. Dynamic model and path specification

This section presents the dynamic equations of multiple robotic arms moving a common object along a specified geometric path. Detailed derivation procedures for these equations can be found in our previous work [22].

2.1. Kinematic analysis

Fig. 1 shows a case of m coordinated rigid robot arms handling a common rigid object. It has been assumed that the object is grasped firmly so there are no relative motions between the object and each of the arm end-effectors during the entire process of manipulation. The arms and the common object form a closed-chain system of rigid bodies. Let n_i represent the number of joint variables of arm i, and $\theta_i = (\theta_1^i, \ldots, \theta_{n_i}^i)^{\mathrm{T}}$ be the corresponding joint variable vector, $i = 1, \ldots, m$. According to the *Kutzbach-Crubler criterion* [15] there are only p independent motion variables among the total joint variables, where p is given by

$$p = \sum_{i=1}^{m} n_i - 3(D-1)(m-1), \quad D = \begin{cases} 2 & \text{for spatial arm systems,} \\ 3 & \text{for planar arm systems.} \end{cases} \tag{1}$$

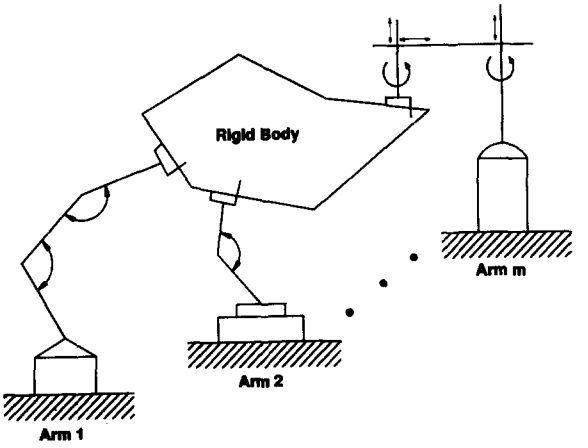


Fig. 1. Common object manipulation by coordinated multiple arms.

Therefore, the motion of the system can be described completely by a generalized coordinate vector $q = (q_1, \ldots, q_p)^T$ as,

$$\theta_i = \theta_i(q), \quad i = 1, \ldots, m.$$

The Jacobian matrices of θ_i can be calculated by,

$$\dot{\theta}_i = J_i \dot{q}, \; J_i = \frac{\partial \theta_i}{\partial q} = \left(J_{jk}^i\right)_{n_i \times p}, \; J_{jk}^i = \frac{\partial \theta_j^i}{\partial q_k}, \quad i = 1, \ldots, m, \; j = 1, \ldots, n_i, \; k = 1, \ldots, p.$$

2.2. Dynamic equations

The dynamic model of the multiple arm system can be developed using *Hamilton's Principle*. The following equations have been obtained for the system [22],

$$M(q)\ddot{q} = f(q, \tau) - C(q, \dot{q})\dot{q} - \gamma(q)\dot{q} - k(q), \tag{2}$$

where

$$\{f(q, \tau), \gamma(q), k(q)\} = \sum_{i=1}^{m} J_i^T(q)\left\{\tau_i, B_i J_i(q), \frac{\partial g_i}{\partial \theta_i}\right\}, \quad \tau_i = \left(\tau_1^i, \ldots, \tau_{n_i}^i\right)^T,$$

$$C(q, \dot{q}) = M_D(q, \dot{q}) - \tfrac{1}{2}M_D^T(q, \dot{q}), \quad M_D(q, \dot{q}) = \sum_{i=1}^{m} \frac{\partial M}{\partial q_i}\dot{q}e_i^T, \quad M(q) = \sum_{i=1}^{m} J_i^T M_i(\theta_i(q)) J_i,$$

in which e_k is the kth unit vector, i.e., the kth element is one, all others are zero, τ_i is the joint actuator torque/force vector, M_i the inertia matrix, B_i the friction coefficient matrix, and g_i the potential energy of arm i, $i = 1, \ldots, m$. Terms $f(q, \tau)$, $C(q, \dot{q})$, $\gamma(q)$ and $k(q)$ represent the effects of joint torque/force, Coriolis/centrifugal forces, friction and gravity, respectively. Usually $B_i = \text{diag}(b_1^i, \ldots, b_{n_i}^i)$ and b_j^i is the friction coefficient of joint j of arm i. It has been assumed that joint torque/force τ_i is bounded by

$$\tau_i^{\min}\left(\theta_i, \dot{\theta}_i\right) \le \tau_i \le \tau_i^{\max}\left(\theta_i, \dot{\theta}_i\right), \quad i = 1, \ldots, m. \tag{3}$$

2.3. Path specification

Assume the common object held by the coordinated arms is required to move along a specified geometric path $h(s)$, where $s_0 \le s \le s_f$ is a scalar called path parameter. In this case by assigning a time history to the path parameter s we can determine the motion of the common object and thus the trajectory of the generalized coordinate vector q (the problems of multiple configurations and singularity can be solved during the path planning stage and will not be discussed here). This can be written as,

$$q(t) = q(s(t)) = h(s(t)), \quad s_0 \le s(t) \le s_f, \tag{4}$$

and correspondingly,

$$\dot{q} = h_s \dot{s}, \quad \ddot{q} = h_{ss}\dot{s}^2 + h_s \ddot{s},$$

where $\dot{s}$ and $\ddot{s}$ are called *pseudo-velocity* and *pseudo-acceleration*, respectively. Also, h_s and h_{ss} denote the first and second order partial derivative of h with respect to s, respectively.

Obviously, the motion of the common object and correspondingly $(q, \dot{q}, \ddot{q})$ will be determined once the time history of $(s, \dot{s}, \ddot{s})$ is known. Hence $(s, \dot{s}, \ddot{s})$ has characterized the motion or trajectory of the coordinated multiple arms along the given geometric path $h(s)$. From $q(t)$ the motion of each individual arm can then be determined using $\theta_i(t) = \theta_i(q(t))$. Therefore, the problem of trajectory planning for the coordinated multiple arms becomes the task of finding path parameter s as a function of time.

The dynamic equations along the given path $h(s)$ can be written in terms of the path parameter s, pseudo-velocity $\dot{s}$ and pseudo-acceleration $\ddot{s}$ as,

$$a(s)\ddot{s} + b(s)\dot{s}^2 + \gamma(s)\dot{s} + c(s) = f(s, \tau), \tag{5}$$

where

$$a(s) = M(h)h_s, \quad b(s) = M(h)h_{ss} + C(h, h_s)h_s,$$
$$c(s) = k(h), \quad r(s) = r(h), \quad f(s, \tau) = f(h, \tau).$$

3. Formulations of optimum trajectory planning

The problem of optimum trajectory planning along a specified geometric path is to determine the time history of the path parameter (i.e., s and $\dot{s}$) that will allow the coordinated multiple arms to move from their initial state to a desired state with a minimum cost under the given dynamic constraints. To formulate the trajectory planning problem in a form that is convenient for the cell-to-cell mapping method to be developed in the next section, we introduce the following state-variable trajectory representation:

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = u(t), \tag{6}$$

where $x_1 = s(t)$ and $x_2 = \dot{s}(t)$ are two state variables and $u(t) = \ddot{s}(t)$ is control.

The general form of the performance index for trajectory planning is defined as,

$$\min_{u(t)\in U} J(u(t)) = \int_0^{t_f} \sum_{i=1}^{m} \mathscr{F}_i\left(\theta_i, \dot{\theta}_i, \ddot{\theta}_i, \tau_i\right) \mathrm{d}t, \tag{7}$$

where t_f is the trajectory execution time, $\mathscr{F}_i$ represents the cost function for arm i, and U is the set of admissible controls. In this paper, we will only discuss three optimum trajectory planning problems, i.e., minimum-time, minimum-energy and minimum-jerk trajectories. However, the method and procedures of optimum trajectory planning discussed here are still valid for other optimum trajectory planning

problems. Note that for a single robotic arm system, if an inappropriate cost function is selected the corresponding minimum trajectory may exhibit poor behavior during motion execution [12].

3.1. Minimum-time trajectory

The minimum-time trajectory planning problem has been studied extensively over the last few years [3,16,13,19,5]. In manufacturing tasks, minimum motion execution is obviously desired for better productivity. A minimum-time trajectory can be achieved by making all the cost function $\mathscr{F}$'s have the same constant value, so that $J(u(t)) = t_f$. Using *Pontryagin* maximum principle, it has been found that the control structure for optimum trajectory generation is that of generalized bang-bang control [3].

The lower and upper bounds for generalized bang-bang control can be found from constraint (3) and Eq. (5),

$$\alpha(x_1, x_2) \le u(t) = \ddot{s} \le \beta(x_1, x_2), \tag{8}$$

where both $\alpha(x_1, x_2)$ and $\beta(x_1, x_2)$ can be determined using a modified linear programming method with the pseudo-acceleration as the objective function. Generalized bang-bang control requires that during any time interval the optimum control must be

$$u(t) = \alpha(x_1(t), x_2(t)) \quad \text{or} \quad u(t) = \beta(x_1(t), x_2(t))$$

Therefore, the problem of time-optimal trajectory planning is reduced to determining the time instances at which the pseudo-acceleration of the path must switch from one bound to the other, i.e., *switch points*. Using this special control structure, it is simple and very efficient to solve the planning problem using the cell mapping method. A detailed description of the procedure for finding the minimum-time trajectory using this method can be found in our previous work [22].

3.2. Minimum-energy trajectory

A minimum-energy trajectory is commonly desired in environments where a limited power supply is available, such as robotic motions for Lunar or Martian tasks in space exploration. In addition, repetitive robotic operations over long time periods, or heavy duty materials handling such as mining transportation tasks for local resource utilization, are cases where a minimum-energy trajectory may reduce the energy consumption level significantly. A minimum-energy trajectory can be obtained by using some energy measure as the cost function, e.g.,

$$\mathscr{F}_i = \dot{\theta}_i^{\mathrm{T}} Q_i \dot{\theta}_i + \tau_i^{\mathrm{T}} R_i \tau_i, \tag{9}$$

where Q_i and R_i are non-negative matrices. The two cost terms reflect kinetic energy and power consumed by arm i, respectively. Clearly, the first term can easily be expressed in terms of trajectory variables s and $\dot{s}$. However, the expression for the second term is much more complicated, since in general, torque τ_i cannot be uniquely determined from $(s, \dot{s}, \ddot{s})$. This is a special problem for multiple arms and will be discussed in detail later. Note that the motion execution time t_f will be treated as fixed in this case.

3.3. Minimum-jerk trajectory

It has been found that the jerk (the third derivative of joint variables) of the desired trajectory adversely affects the efficiency of the arm motion control algorithm [10]. Experimental results have indicated that joint position errors increase when jerk increases [11]. Therefore jerk should be minimized in order to accomplish motion tasks accurately and smoothly. This planning strategy may also be useful

for manipulation tasks involving large and fragile components in construction operations. To specify a minimum-jerk trajectory, the cost functions have to be chosen as,

$$\mathscr{F}_i = \dddot{\theta}_i^{\mathrm{T}} S_i \dddot{\theta}_i,$$

where S_i is a positive matrix. In the single robot arm case, the problem has been solved by [10] using an optimal control formulation without considering dynamic force/torque constraints. However, for the coordinated multiple arms case, the same formulation cannot be used due to its complexity and nonlinear dynamic constraints (3) and (5).

Usually, the calculation of the third derivatives is very complicated and time consuming. To avoid involving the third derivatives directly, we will approximate them by the second derivatives instead. Let $t_0 = 0 < t_1 < \ldots < t_k < \ldots < t_n = t_f$ be a discretization of the motion execution period. Then the cost functions for the minimum-jerk trajectory can be replaced by

$$\mathscr{F}_i(t_k) = \left(\ddot{\theta}_i(t_k) - \ddot{\theta}_i(t_{k+1})\right)^{\mathrm{T}} S_i \left(\ddot{\theta}_i(t_k) - \ddot{\theta}_i(t_{k+1})\right). \tag{10}$$

Now it is relatively easily to express $\mathscr{F}_i$ in terms of x_1, x_2 and u. However, the value of the cost functions at t_k in this case also depends on the control to be selected at the next time instance t_{k+1} (since $\ddot{\theta}_i$ is a function of u), thus the cost at a given state cannot be calculated only based on the control chosen at that state. This problem can be solved by introducing one more state variable in trajectory representation, i.e.,

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = x_3, \quad \dot{x}_3 = u(t), \tag{11}$$

where the new trajectory state variable and control are defined as $x_3 = \ddot{s}(t)$ and $u(t) = \dddot{s}(t)$, respectively, and x_3 must meet constraint (8). With the new state variables, the value of cost functions at t_k now can be found by using the current state and control information only, i.e., $\mathscr{F}_i(t_k) = \mathscr{F}_i(x_1, x_2, x_3, u/t = t_k)$. The actual computation procedure can easily be constructed from Eq. (4). Correspondingly, the performance index is modified as,

$$\min_{u(t)} J(u(t)) = \sum_{k=0}^{n-1} \sum_{i=1}^{m} \mathscr{F}_i(t_k). \tag{12}$$

4. Optimum trajectory search with cell mapping

The basic concept of the cell-to-cell mapping method and the procedure for using this approach to plan optimum trajectories are described in this section. A simple trajectory search algorithm that uses cell-to-cell mapping is discussed.

4.1. Cell-to-cell mapping

The cell-to-cell mapping method was developed by Hsu [6,7] in the early 80's for nonlinear dynamic analysis. The basic idea is to consider a state space not as a continuum, but as a collection of state cells with each cell representing a state entity. Let x_i, $i = 1, 2, \ldots, N$, be a state variable of the state space, and let the coordinate axis of state variable x_i be divided into a large number of intervals of uniform interval size h_i. The interval z_i along the x_i-axis is defined to be the one that contains all x_i satisfying

$$\left(z_i - \tfrac{1}{2}\right)h_i \le x_i < \left(z_i + \tfrac{1}{2}\right)h_i,$$

where z_i is an integer. Such an N-tuple $(z_1, z_2, \ldots, z_N)$ is then called a *cell* and is denoted by z. The *cell state space* $S \subset Z^N$ is the collection of all such cells. We will assume S is finite with cardinality N_s.

Having introduced a cellular structure to the state space, we can approximate a dynamic system with continuum state space,

$$\dot{x} = f(x, u, t), \quad x \in R^N, \quad u \in R^M, \tag{13}$$

by a dynamic system over the discrete cell state space,

$$z(n+1) = C(z(n), u(n)), \quad z \in S \subset Z^N, \quad u \in U \subset Z^M, \tag{14}$$

where $C(\cdot, \cdot)$ is called *cell-to-cell mapping*. Control u has also been discretized in the cell-to-cell mapping. The set of admissible discrete controls U will be chosen as a finite set with at most N_u different values. Note that U is allowed to vary from cell to cell. We call $z(n)$ and $z(n+1)$ the *domain cell* and the *image cell*, respectively. Clearly, a domain cell may have at most N_u possible image cells.

To make the cell mapping method more efficient, we also need to discretize the one-step value of the cost functions [8]. Since each cell may have up to N_u values for its one-step cost, there could be at most $N_u \times N_s$ different one-step cost levels. Let N_w represent the number of one-step cost levels selected. Usually, N_w is far less than its upper bound $N_u \times N_s$.

For optimum trajectory planning with the cell mapping method, it is very easy to convert (6) or (11) to a cell-to-cell mapping equation. Note that cells with $\alpha > \beta$ are inadmissible under the given torque constraints for the given path specification and can be classified as a single *dead* or *sunk* cell. This will guarantee that the cell space to be considered has a finite number of cells. The admissible control set U for each cell will be determined according to the values of α and β calculated at that cell.

4.2. One-step cost calculation

In minimum-energy trajectory planning, individual joint force/torque τ_i for each arm cannot be fully determined from trajectory variables x_1 and x_2. Thus, a quadratic programming approach has been used at each cell to find the force/torque distribution and to achieve a minimum-energy trajectory. In other words, the one-step cost is calculated by,

$$\mathscr{F}(z(i), u(i)) = \min_{\tau_1, \ldots, \tau_m} \sum_{i=1}^{m} \left(\dot{\theta}_i^{\mathrm{T}} Q_i \dot{\theta}_i + \tau_i^{\mathrm{T}} R_i \tau_i \right), \tag{15}$$

subject to constraints (3) and (5) at the given cell $z(i)$ with control $u(i)$. Once $\mathscr{F}$ is determined, it will be rounded to one of the N_w discretized one-step cost levels.

The calculation of one-step cost for minimum-jerk trajectory planning is straightforward using cost function approximation (10). Again, once $\mathscr{F}(z(i), u(i))$ is obtained at each cell, it is rounded to a discretized cost level. Note that we now have to deal with a three-dimensional instead of a two-dimensional cell space as for the minimum-energy problem.

4.3. Cell-space search

Based upon the cellular structure of state space and the discretization of control and cost functions, an optimum trajectory planning problem associated with the continuum dynamic system (6) or (11) can be transferred into a search problem that relates to the corresponding discrete cell-to-cell mapping (14) over the cell space. To see this, let the cell space be partitioned into three parts, i.e., $S = \Theta \cup S_d \cup S_u$, where Θ is the set of target cells, S_d the set of all cells from which the target set cannot be reached without violating the given path or force/torque constraints, and S_u the set of all other cells. Obviously, S_d contains the dead cell. Both S_d and S_u are not known before the completion of the trajectory search.

The detailed search process and data structure for information tracking is discussed in [8,22]. Let $p(N_s)$ be an upper bound of the performance index (see the next section for its estimation), $cc(z)$ the accumulated cost of moving optimally from cell z to the target Θ, $nn(z)$ the optimal image cell of z, and $\{w_0, \ldots, w_{N_w-1}\}\sigma$ the set of possible one-step cost levels in ascending order, where σ is the unit cost and w's are integers. The major steps of the search algorithm can be outlined as follows:

Begin
  $W = w_0$; $S_u = S - \Theta -$ {dead cell}; $S_v = \Theta$;
  While $W \le p(N_s)$ **Do**
    $D = \phi$;
    For each cell $z \in S_u$ **Do**
      Find $H(z) = \{z' \mid z' = C(z, u) \in S_v$ and $cc(z') = W - w_i$ for some $u \in U$ and $0 \le i < N_w\}$;
      If $H(z) = \phi$ **Then** go to next cell in S_u;
      Else
        Break ties using discriminating functions if $H(z)$ contains more than one cell;
        Select the cell left in $H(z)$ as $nn(z)$;
        $cc(z) = W$; $D = D \cup \{z\}$;
      End
    End
  $S_u = S_u - D$; $S_v = S_v \cup D$;
  Increase W to the next cost level;
End,

where S_v is the set of all cells whose optimum control has been decided. Also, at the end of the search process $S_d = S - S_v$ corresponds to the set of cells from which the target cannot be reached with the given path or force/torque constraints. An important feature of the search algorithm is its independence to initial conditions, i.e., once the target cell is given, the cell-to-cell mapping search will determine all the optimum trajectories starting from any initial cells by a single search.

The discriminating functions mentioned in the above algorithm are used to break possible ties encounted when different image cells offer the same total optimum cost during the search process. There are various ways to select the discriminating functions [8,22]. For example, we can use the deviation of the image point of the current cell to the center of its image cell as the first discriminating function, and the sum of the accumulated deviations along the optimum trajectory from the current cell to the target as the second. If a tie still exists after testing the two discriminating functions, a random selection can be used to break the ties.

5. Complexity estimation and hierarchical search

Complexity analysis is performed for the search algorithm described in the previous section. To improve the search efficiency and enable the parallel implementation, a hierarchical search scheme is proposed. The complexity of the hierarchical search strategy and the problem of finding an optimum number of cell space divisions is also discussed.

5.1. Upper bound cost and search complexity

The upper bound cost of a trajectory can be estimated by imposing some construction restriction on the cellular structure. Physically, an arm system should always move to a new position/orientation at

each control step. This suggests that the division of cell space and control needs to be refined so that a cell will always be mapped into a new cell with a greater $x_1 = s$ coordinate. In other words, the cell mapping will always evolve along its z_1 coordinate. Using this division policy, a trajectory from any initial cell to the target can consist of at most N_1 cells, where N_1 is the number of divisions along the x_1 axis. In the worst case, each mapping step will cost the maximum one-step-mapping cost w_{N_w-1}, thus the upper bound of the trajectory cost is,

$$p(N_s) = N_1 N_w. \tag{16}$$

Note that $p(N_s)$ is not dependent on the number of divisions along the x_2 or the x_3 axis.

From the search algorithm in the previous section, the number of candidate cells in S_u should become smaller and smaller as the search proceeds. However, the exact number of candidates entering S_v at each cost level cannot be determined. In the extreme case, the search at a given cost level may extend over the whole cell space. Therefore, the worst-case computation complexity for the search algorithm can be estimated as $N_s p(N_s) T_c$, where T_c represents the cell-to-cell mapping search computation time required to evaluate a single candidate cell. T_c is bounded by $T_c \leq c_\alpha + c_\beta + c_{\mathscr{F}} N_u$, where c_α, c_β and $c_{\mathscr{F}}$ are the computation times of α, β, and one-step cost $\mathscr{F}$, respectively.

5.2. Hierarchical search strategy

The computational efficiency of the cell-mapping search process can be improved by a hierarchical implementation. The basic idea is to divide one optimum trajectory planning problem into several small ones along a given path. This division leads the original planning problem into a two-level decision problem: a multivariable optimization problem in the higher level and several independent optimum trajectory planning problems at the lower level as shown in Fig. 2. All individual trajectory planning problems at the lower level are solved using the same search algorithm described in the previous section. Independence at the lower level allows a parallel execution of the search process.

Consider the minimum-energy planning problem specifically. We divide the original total cell space into d small sub-cell-spaces along the x_1 axis with each containing $(N_1/d)N_2$ cells, where N_2 is the number of divisions along the x_2 axis. Let Θ_i be the target cell assigned for sub-cell-space i by the higher level, i.e., $\Theta_i = (i(N_1/d), y_i)$, where y_i is an integer representing the pseudo-velocity of the target cell, $i = 1, \ldots, d$. Θ_i also serves as the search starting cell of sub-cell-space $i+1$, except for $\Theta_d = \Theta$, which is the global target cell. The task of the higher level is to find the optimum pseudo-velocity vector $(y_1, y_2, \ldots, y_d)$ such that the sum of the individual costs submitted by d local cell-to-cell mapping searches at the lower level will yield the global minimum cost for the total cell space. In other words, the

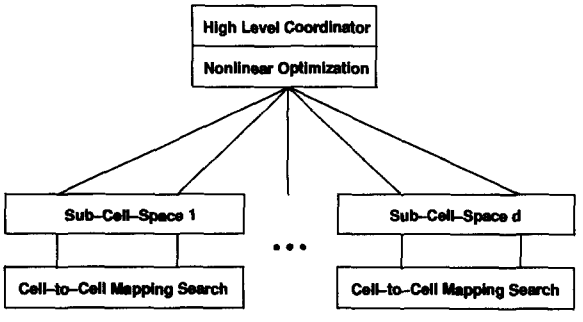


Fig. 2. Hierarchical search structure for cell mapping search.

optimum pseudo-velocity vector will link the optimum sub-trajectories found independently by d local cell-to-cell mapping searches to form a global optimum trajectory in the total cell space.

The hierarchical search algorithm owes much of its efficiency to the fact that once the target cell is given, the cell-to-cell mapping search will determine all optimum trajectories leading to the target cell, irrespective of the initial starting cell. This means that one only needs to look up the relevant tables to obtain the minimum cost and optimum trajectory from any cell to the target cell without executing a new search process each time. The cell mapping searches in the lower level can be carried out either sequentially or in parallel. The efficiency of the hierarchical search strategy in these two cases can be estimated using the following worst-case complexity analysis.

5.3. Hierarchical search evaluation

Let T_a and T_b respectively denote the complexities of adding two variables and selecting a new velocity vector for searching at the higher level (many optimization methods can be used for this purpose). In the worst case, the higher level exhausts all possible values of pseudo-velocity vector, i.e., all N_2^{d-1} vectors (since Θ_d is fixed) and thus its corresponding worst-case complexity can be calculated as:

$$T_h = N_2^{d-1}(T_b + (d-1)T_a). \tag{17}$$

In the lower level, each sub-cell-space has N_2 possible target cells with the exception of the last one which receives the global target cell as its only target cell. Thus, according to discussions in the previous subsection, the worst-case complexity of the lower level can be estimated as,

$$T_s = ((d-1)N_2 + 1)\frac{N_1^2 N_2 w_{N_w-1}}{d^2}T_c, \quad T_p = N_2\frac{(N_1N_2)^2 w_{N_w-1}}{d^2}T_c, \tag{18}$$

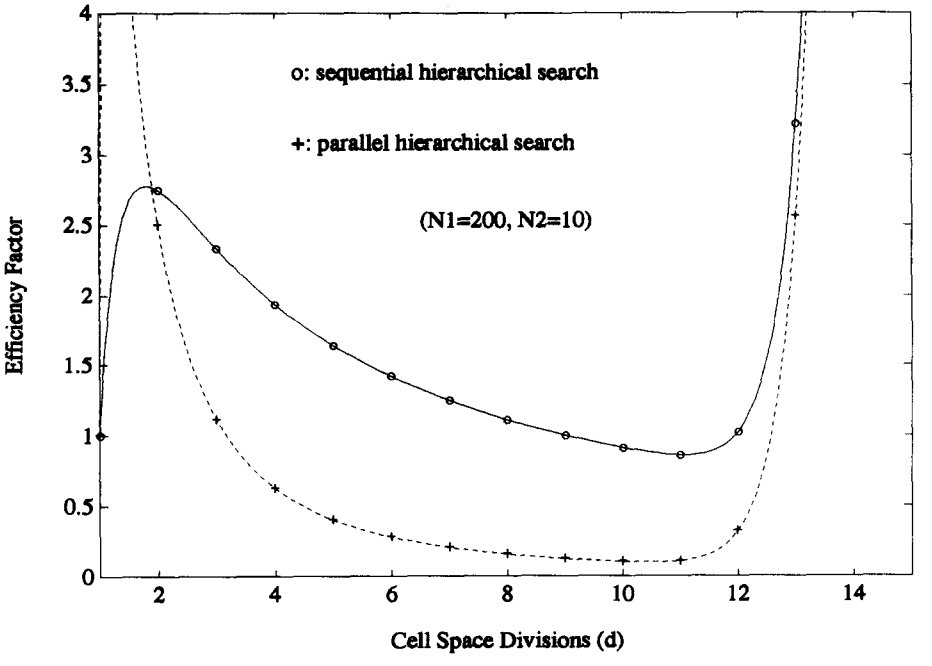


Fig. 3. Efficiency of the hierarchical search strategies.

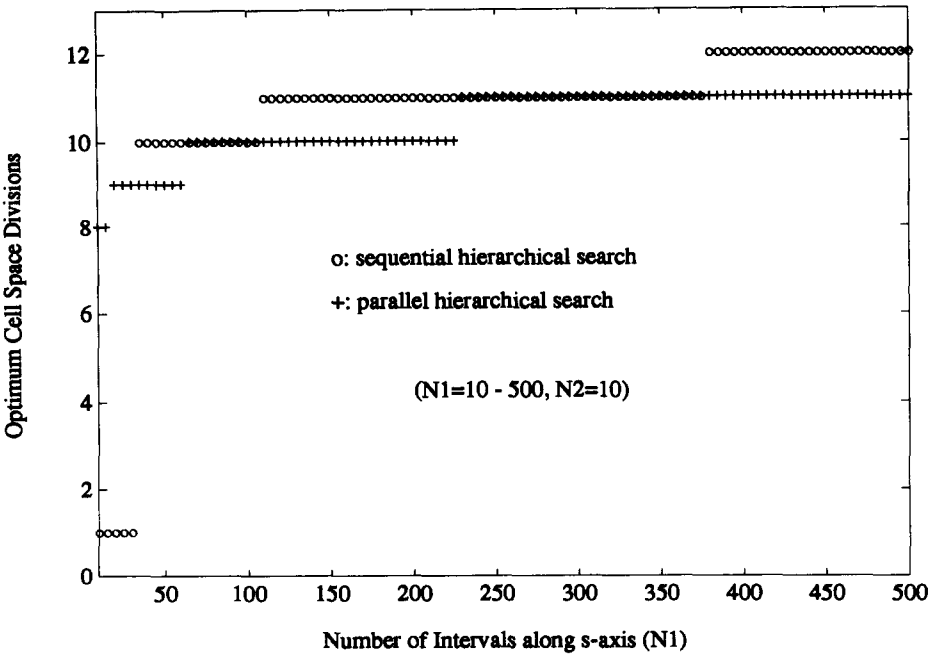


Fig. 4. Optimum number of cell space divisions.

for the sequential and parallel searches, respectively. Correspondingly, the efficiency factor of the hierarchical search strategy, defined as the ratio of complexities of the hierarchical and the single cell space searches, can be determined as,

$$\eta_s = \frac{T_h + T_s}{N_1^2 N_2 w_{N_2-1} T_c}, \quad \eta_p = \frac{T_h + T_p}{N_1^2 N_2 w_{N_w-1} T_c}, \tag{19}$$

for the sequential and parallel searches, respectively. Fig. 3 gives η_s and η_p as a function of d for $N_1 = 200$, $N_2 = 10$, $w_{N_w-1} = 10$; $T_a/T_c = 10^{-12}$ and $T_b/T_c = 10^{-6}$. Note that the sequential search is not efficient in this case except when $d = 10$ or 11, where $\eta_s = 0.9125$ or 0.8597, respectively.

The optimum division of the total cell space is found by selecting d so that η_s or η_p obtains its minimum value. For example, assuming the same data used in Fig. 3 except $N_1 = 100$, one can find $d_{\text{optimum}} = 10$ for both the sequential and parallel searches. The corresponding $\eta_s = 0.92$, i.e., only 8% of the computation time can be saved by the sequential hierarchical search; however, more significantly, $\eta_p = 0.11$, i.e., an 89% saving by the parallel hierarchical search. Fig. 4 shows d_{optimum} as a function of N_1.

For minimum-jerk trajectory planning, a similar analysis can be performed and the corresponding complexity estimation can be obtained by replacing N_2 with $N_2 N_3$, where N_3 is the number of divisions along the x_3 axis.

6. Conclusions

Several cell-mapping based search schemes have been proposed in this paper for planning optimum trajectories of coordinated multiple arm systems using general performance indices. In particular,

minimum-energy, minimum-jerk, and minimum-time trajectory planning formulations have been discussed. A simple cell-space search algorithm as well as its hierarchical implementation has been proposed. The worst-case computational complexity of the hierarchical search strategy is estimated and compared with that of the basic search algorithm. The results show a large improvement in search efficiency with the hierarchical search method, especially when the search over each sub-cell-space is executed in parallel. For the example calculated in section 5.3, an 89% saving in the worst-case computation time was achieved using the parallel hierarchical search.

The major advantages of using the cell mapping method for optimum trajectory planning includes its simplicity and applicability to a wide range of problem formulations. A special feature of this search method is its ability to find all optimum trajectories for all possible initial conditions through a single search. Conventional methods based on optimum control formulations usually involve different, complicated equations and treatments for different planning problems [10,16]. However, the construction of cellular space and cell-to-cell mapping for a particular dynamic system is still very tedious. For example, many trial-and-errors are needed before a proper cell size, control discretization, and number of one-step cost levels can be determined. Computer-aided-design software packages which facilitate the applications of the cell mapping method must be developed.

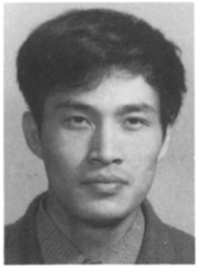
7. Acknowledgement

This work was supported in part by the Arizona Mining and Mineral Resources Research Institute under Grant #1134204.

8. References

[1] J.E. Bobrow, S. Dubowsky and J.S. Gibson, Time-optimal control of robotic manipulators along specified paths, *International Journal of Robotics Research* 4 (3) (1985) 3–17.
[2] Yaobin Chen and Alan A. Desrochers, Structure for minimum-time control law for robotic manipulators with constrained paths, *Proc. IEEE Conference on Robotics and Automation* (1989) 971–976.
[3] Yaobin Chen and Alan A. Desrochers, On the time-optimal control law for robotic arms moving a common object along specified path, *IEEE Tran. Automat. Contr.* AC-37 (1992).
[4] Yao-Chon Chen, On the structure of time-optimal controls for robotic manipulators, *IEEE Tran. Automat. Contr.* AC-29 (1989) 115–116.
[5] Y.P. Chien and Qing Xue, Trajectory planning for coordinately operating robots, *AI EDAM* 4 (3) (1990) 165–177.
[6] C.S. Hsu, A theory of cell-to-cell mapping dynamical systems, *J. Applied Mechanics* 47 (1980) 928–939.
[7] C.S. Hsu, A generalized theory of cell-to-cell mapping for nonlinear dynamical systems, *J. Applied Mechanics* 48 (1981) 634–842.
[8] C.S. Hsu, A discrete method of optimal control based upon the cell state space concept, *J. of Optimization Theory and Applications* 46 (1985) 547–569.
[9] Musa K. Jouaneh, Zhixiao Wang and David A. Dornfeld, Trajectory planning for coordinated motion of a robot and a position table, Part I and II, *IEEE Transactions on Robotics and Automation* 6 (6) (1990) 735–745.
[10] K.J. Kyriakopoulos and G.N. Saridis, Minimum jerk path generation, *Proc. IEEE International Conf. on Robotics and Automation*, Philadelphia (1988) 364–369.
[11] M.B. Leahy and G.N. Saridis, Compensation of unmodeled PUMA manipulator dynamics, *Proc. IEEE International Conference on Robotics and Automation* (1987) 151–156.
[12] M.S. Mujtaba, Discussion of trajectory calculation methods, in: T.O. Binford et al., eds., *Exploratory Study of Computer Integrated Assembly Systems* (Standford University, AI Laboratory, 1977) AIM 285.4.
[13] F. Pfeiffer and R. Johanni, A concept for manipulator trajectory planning, *IEEE J. Robotics Automat. Contr.* RA-3 (1987) 2.
[14] G. Sahar and J.M. Hollerbach, Planning of minimum-time trajectories for robot arms, *Int. J. of Robotics Research* 5 (1986) 3.
[15] Shigley, J.E., *Kinematic Analysis of Mechanics* (McGraw-Hill, New York, 1969).

[16] K.G. Shin and N.D. McKay, Minimum-time control of robotic manipulators with geometric path constraints, *IEEE Transactions on Automatic Control* AC-30 (3) (1985) 531–541.
[17] K.G. Shin and N.D. McKay, A dynamic programming approach to trajectory planning of robotic manipulators, *IEEE Transactions on Automatic Control* AC-31 (1986) 491–500.
[18] K.G. Shin and Qin Zheng, Minimum time trajectory planning for dual robot system, *IEEE Conf. on Decision and Control*, Tempa, FL (Dec. 1989) 2506–2511.
[19] S. Singh and M.C. Leu, Optimal trajectory generation for robotic manipulators using dynamic programming, *Journal of Dynamic Systems, Measurement, and Control* 109 (June 1987).
[20] Jean-Jacques E. Slotine and Hyun S. Yang, Improving the efficiency of time-optimal path following algorithm, *IEEE Transactions on Robotics and Automation* 5 (1989) 118–124.
[21] W.H. Zhu and Ming C. Leu, Planning optimal robot trajectory by cell mapping, *Proc. IEEE Conference on Robotics and Automation* (1990).
[22] Fei-Yue Wang and Bing Pu, Time-optimal trajectory generation for coordinated robot arms handling a common object using the cell-to-cell mapping method, *Journal of Optimization Theory and Applications* 84 (1) (1995).
[23] Fei-Yue Wang and Paul J.A. Lever, An intelligent robotic vehicle system for lunar/martian resource assessment, in: Y.F. Zheng, ed., *Advanced Mobile Robots: Theory and Applications – An International Perspective* (World Scientific Publishers, New York, 1993).
[24] P.K.C. Wang, A method for approximating dynamical processes by finite-state system, *Int. J. Control* 8 (1968) 285–296.



Fei-Yue Wang received his Ph.D. in Computer and Systems Engineering from Rensselaer Polytechnic Institute in 1990 and since then is an Assistant Professor of the Systems and Industrial Engineering Department at the University of Arizona. From 1986 to 1990 he was with the RPI/NASA Center for Intelligent Robotic Systems for Space Exploration. In his previous work in mechanics and applied mathematics, he contributed significantly to the theoretical development of shells, plates, planes, 3D elasticity, and micropolar elasticity. He is the co-author of the book *Coordination Theory of Intelligent Machines: Applications in Intelligent Robotic Systems and CIM Systems* and has published more than 40 refereed journal articles and book chapters, and about 50 conference papers and technical reports in the areas of applied mathematics, mechanics, mechatronics, artificial intelligence, control theory, communication, robotics and automation, computer-integrated manufacturing, fuzzy logic, neural networks, and intelligent machines. Dr. Wang is a member of Sigma Xi, IEEE, ACM, ASME, and NCoSE.

Paul Lever received a B.Sc. degree in Mining Engineering from the University of Witwatersrand, Republic of South Africa, in 1982, and the M.Sc. and Ph.D. degrees in Mining Engineering from the Colorado School of Mines, Golden, in 1987 and 1991 respectively. He is currently an Assistant Professor in the Department of Mining and Geological Engineering at the University of Arizona in Tucson. He is the author of more than 20 refereed journal articles, conference papers and technical reports in the areas of mine hoisting systems, machine learning, sensor based intelligent decision support system and mine automation and robotics. His research interests include robotics systems, intelligent machines and control, artificial intelligence, sensor fusion and specifically intelligent control in unstructured and dynamic environments (mining). Dr. Lever is a member of the AIME (Society of Mining Engineers), and IEEE.