controller bandwidth. Meanwhile, since the measurement system is mounted on the hub, this controller configuration can, in principle, be extended to multiple-link robots.

VI. Conclusions

Position control of a single-link flexible beam has been investigated by using an output feedback controller with a shaft encoder and a slope sensor used as feedback signals. Experimental results show that the system responses are in good agreements with simulation results. The performance of the overall system is satisfactory, and the position controller is robust to parameter variations.

Acknowledgment

The authors wish to thank T. S. Chang for designing part of the sensor system.

References

[1] W. J. Book, "Modeling, design and control of flexible manipulator arms," Ph. D. dissertation, MIT, Dept. of Mech. Eng., Apr. 1974.
[2] W. J. Book and M. Majette, "Controller design for flexible, distributed parameter mechanical arms via combined state space and frequency domain techniques," *ASME J. Dyn. Syst., Meas., Cont.*, vol. 105, no. 4, pp. 245-254, Dec. 1983
[3] R. H. Cannon, Jr., and E. Schmitz, "Initial experiments on the end-point control of a flexible one-link robot," *Int. J. Robotics Res.*, vol. 3, no. 3, pp. 62-86, Fall 1984.
[4] H. Kanoh and H. G. Lee, "Vibration control of one-link flexible arm," in *Proc. 24th IEEE CDC* (Ft. Lauderdale, FL. Dec. 1985), pp. 1172-1177.
[5] H. Kanoh, S. Tzafestas, H. G. Lee, and J. Kalat, "Modelling and control of flexible robot arms," in *Proc. 25th IEEE CDC* (Athens, Greece, Dec. 1986), pp. 1866-1870.
[6] G. D. Martin, "On the control of flexible mechanical system," Ph. D. dissertation, Stanford Univ., Dept. of Elec. Eng., May 1978.
[7] Y. H. Lin and J. G. Lin, "An improved output feedback control of flexible large space structures," in *Proc. 19th IEEE CDC* (Albuquerque, NM, Dec. 1980), pp. 1248-1250.
[8] H. J. Woltring, "Single and dual-axis lateral photodetectors of rectangular shape," *IEEE Trans. Electron Devices,* vol. ED 22, pp. 581-586, Aug. 1975.
[9] W. J. Wang, S. S. Lu, and C. F. Hsu, "Output feedback control of a flexible robot arm," presented at the 25th IEEE CDC, Athens, Greece, Dec. 6-9, 1986.

Planning Time-Optimal Robotic Manipulator Motions and Work Places for Point-to-Point Tasks

S. DUBOWSKY AND T. D. BLUBAUGH

Abstract—**High productivity requires that manipulators perform complex tasks quickly. Recently developed optimal control algorithms enable manipulators to move quickly, but only for simple motions. A method is presented here which combines simple time-optimal motions in an optimal manner to yield the minimum-time motions for an important class of complex manipulator tasks composed of point-to-point moves, such as assembly, electronic component insertion, and spot welding. This method can also be used to design manipulator actions and work places so that tasks can be completed in minimum time. The method has been implemented in a computer-aided design (CAD) software system; several examples are presented. Experimental results show the method's validity and utility.**

Manuscript received July 9, 1986; revised August 19, 1988. This work was supported by the NASA Langley Research Center under Grant NAG-1-489 and by the General Motors Corporation in the form of a Fellowship to T. D. Blubaugh. Part of the material in this communication was presented at the 24th IEEE Conference on Decision and Control (CDC), Ft. Lauderdale, FL, Dec. 11-13, 1985.

S. Dubowsky is with the Massachusettes Institute of Technology, Cambridge, MA 02139.

T. D. Blubaugh is with the General Motors Corporation, Warren, MI 44482.

IEEE Log Number 8825232.

I. Introduction

High productivity requires manipulators that are capable of performing complex tasks in minimum time. Current industrial manipulators, however, are not controlled to achieve minumum-time motions, nor are their tasks and work places structured so as to decrease the required task time. The research community has for some time considered control algorithms to enable robotic manipulators to perform movements as quickly as possible within the limits imposed by their physical characteristics [1], [2].

A recently developed algorithm can find a manipulator's minimal-time motion along a specified path. The algorithm includes the effects of the manipulator's nonlinear dynamics and of the manipulator's actuator saturation characteristics. These characteristics may be expressed as arbitrary functions of the manipulator's state [3]. This algorithm, which has been shown to be rigorously optimal and computationally efficient [4], does not require the usual multi-parameter iterations common in optimal control. This algorithm has been extended to six-degree-of-freedom manipulators and implemented in a computer-aided design package OPTARM [5]. The effects of additional constraints, such as payload and end-effector limitations, have been added to the method [6]. The algorithm also forms the basis for methods to optimize the paths of manipulators [7], including the problem of avoiding obstacles [8]. However, the minimum time control algorithms developed to date are practical only for relatively simple motions.

This communication presents a method for combining simple optimal motions in an optimal manner that makes it possible to plan an important class of complex manipulator tasks so that they can be performed in minimum time. Those tasks are composed of moves between points, where the manipulator comes to a full stop at each point. Mechanical assembly, insertion of electronic components, and spot welding operations are common examples.

The order in which manipulators must go to a number of positions and return to a starting point is not critical in many industrial operations, such as in most spot-welding tasks. However, that order can significantly affect the total time required to perform the task. If there are n points in the operation then there will be $(n - 1)!$ possible combinations; finding the order, or tour, with the shortest possible time by trial and error is not practical.

This problem is nearly identical to the well-known Traveling Salesman Problem (TSP) in mathematics. There the objective is to find the tour of all cities in a set with the shortest total distance. A great deal of study has been done to develop algorithms that avoid evaluating all $(n-1)!$ tours in finding the shortest tour [9]-[15]. Some algorithms give exact solutions and some give approximate solutions [10].

In this work an exact tour building algorithm using a branch-and-bound technique based on Little's approach is used to solve the TSP [9], [15]. However, the results shown in this communication do not depend on the use of this algorithm; other TSP algorithms may be selected based on specific computational considerations.

Time must replace distance as the parameter to be optimized in the application of TSP algorithms to the minimum time planning of manipulator tasks. Also, in manipulator problems the minimum travel time from point i to point j will not be equal to the minimum time from j to i because of such factors as gravity and nonlinear system characteristics. This asymmetric nature distinguishes the robot problem from the standard TSP [13].

Finally, many manipulator tasks do require some ordering of the points in a typical operation. For example, a bolt must be inserted in the hole before its nut is put on in assembly. The direct application of TSP algorithms is not possible for these tasks. We show that TSP algorithms can be modified to include such operations. This communication also shows the use of modified TSP algorithms to structure manipulator tasks and design the work environment so that the tasks can be completed in minimum time.

The methods developed in this study have been implemented in a computer-aided design (CAD) software package to enhance its practical use. Several applications of the technique are presented, including experimental results, which demonstrate its effectiveness.

II. The Analytical Method: Building an Optimal Tour

A. Unrestricted Tasks

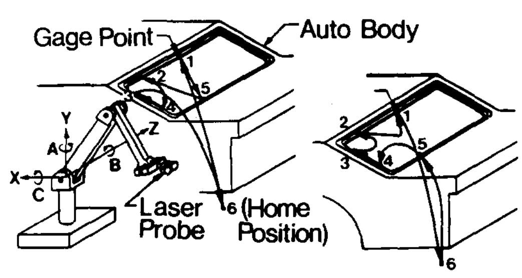
The manipulator planning problem can be formulated directly into a TSP where points in the task may be visited in any order. This section treats the problem of finding the optimal tour for this case. In later sections the method is modified for the more common robotic manipulator applications, where the order of the points is restricted in some measure by the task, and hence, standard TSP formulations cannot be applied directly.

The first step in finding the optimal task tour is to find the optimal times required by all of the possible simple moves which can compose the task. These are most easily shown in the form of an n by n matrix C for an n-point task. The i, j element of C is the time to move from point i to point j. The move from point i to point i is a trivial condition and is excluded from the optimal tour by setting its time to infinity.

The TSP algorithms find the minimum-time paths based on manipulation of a problem's C matrix. For the details of these algorithms the reader is referred to the literature, such as [9], [12]. In this study, Little's algorithm was used to solve the unrestricted problems. It was implemented in Fortran on a relatively small Digital Equipment Corporation PDP-11/44 minicomputer. The method described above was modified using an initial "branch to the right" approach in order to reduce the required memory [9], [16]. The computation times required with this approach were quite reasonable even for this minicomputer system. A typical problem of 55 points requires approximately 1 min of computation time. Fast solutions may be possible using TSP algorithms for different computational environments. It also should be noted that computation time is not critical for this off-line task planning problem.

This method may be applied directly to planning robotic manipulator tasks where the manipulator must come to a stop at each point and may visit the points in any order. The only data required are the values of the individual move times for the C matrix. Ideally, these should be the optimal times obtained using the methods discussed above for simple motions. However, if these approaches are not used, dynamic simulations and experimental timings may be substituted. These should result in improved task performance over *ad hoc* task planning, although the results will be suboptimal. The use of any method to obtain the individual move times may require substantial effort for very large problems. A method to reduce this effort is discussed later in this communication.

The following paragraphs present an example of the method's application to the operation of an automatic robot auto body inspection (ROBI) system, see Fig. 1. The manipulator must carry a laser probe to various positions of an automotive window to check dimensional accuracy. The individual moves were first optimized using the OPTARM program [5], [6]. Then three cases were studied. First, the optimal motions (based on OPTARM) were assembled with an intuitive tour, one which might have been chosen by the system planner using his best judgement for the fastest tour. Second, the task was assembled optimally using estimated individual motion times based on a conventional control (nonoptimal). Third, the task was planned optimally using optimal-move times. The optimal-move times were obtained analytically using OPTARM for a manipulator configuration called ZT6 (see [16]). The resulting C matrix is



(a) (b)

Fig. 1. An unstructured task for optimal planning. (a) Optimal tour. (b) Intuitive tour.

$$\begin{bmatrix} \infty & 0.42 & 0.60 & 0.59 & 0.39 & 0.70 \\ 0.42 & \infty & 0.41 & 0.43 & 0.40 & 0.93 \\ 0.62 & 0.47 & \infty & 0.25 & 0.44 & 0.69 \\ 0.61 & 0.43 & 0.25 & \infty & 0.49 & 0.78 \\ 0.39 & 0.40 & 0.43 & 0.50 & \infty & 0.78 \\ 0.69 & 0.91 & 0.73 & 0.77 & 0.79 & \infty \end{bmatrix}.$$

The C matrix calculated for conventional control is

$$\begin{bmatrix} \infty & 0.64 & 0.87 & 0.86 & 0.67 & 1.96 \\ 0.64 & \infty & 0.40 & 0.48 & 0.65 & 1.85 \\ 0.87 & 0.51 & \infty & 0.26 & 0.69 & 1.58 \\ 0.86 & 0.48 & 0.26 & \infty & 0.61 & 1.49 \\ 0.67 & 0.65 & 0.69 & 0.61 & \infty & 1.81 \\ 1.96 & 1.85 & 1.58 & 1.49 & 1.81 & \infty \end{bmatrix}.$$

The optimal tour under optimal control required a task time of 2.85 s. The intuitive tour with optimal control takes 3.13 s, 9.8 percent longer. These two tours are shown in Fig 1. With moves based on conventional control the optimal tour had a task time of 5.27 s and an intuitive tour took 5.79 s. Again the nonoptimal tour task time was substantially higher (9.9 percent) than the optimal tour. Optimizing the individual moves using OPTARM gave the greatest improvements for this relatively simple problem.

This technique was also demonstrated in laboratory experiments. Fig. 2 shows a relatively simple case where a PUMA 260 mainpulator was used experimentally to simulate a scaled down automotive spot welding operation where five spot welds needed to be placed around the back window of a scaled down automobile. The spot welding was simulated by closing the gripper on the part and pausing for 0.5 s. The paths between points were taken as the most direct ones which avoided contact between the body and the manipulator. Manipulator motions were programmed in VAL. It was not possible to program the individual moves to be optimal because of the restrictions of VAL. Intermediate points were programmed to avoid the temporary transfer tab between points 1 and 2. The individual move times, including "welds," were measured and are contained in the C matrix:

$$\begin{bmatrix} \infty & 18 & 13 & 11 & 14 & 11 \\ 15 & \infty & 16 & 15 & 16 & 12 \\ 12 & 15 & \infty & 11 & 16 & 13 \\ 11 & 15 & 11 & \infty & 14 & 13 \\ 14 & 15 & 15 & 13 & \infty & 11 \\ 17 & 18 & 18 & 19 & 17 & \infty \end{bmatrix}.$$

The optimal tour for this problem is: 6–5–1–4–3–2–6. The optimal tour's measured time was 7.4 s, slightly less than predicted by the TSP calculations because of VAL introduced delays during the initial clocking of individual movements. The setup time for this case was less than 4 h, including all measurements. By contrast, the manually planned intuitive tour of 6–1–2–3–4–5–6 averaged 8.0 s, 8 percent longer than the optimal tour, a rather significant saving. In such a simple problem it was easy to pick a good tour intuitively. This would

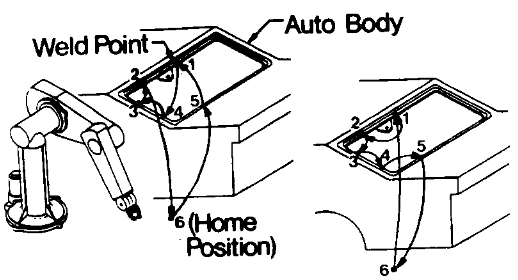


Fig. 2. An experimentally simulated spot welding task. (a) Optimal tour. (b) Intuitive tour.

not have been true in cases where dozens of spot welds were required.

B. Ordered or Semi-Continuous Tasks

Tasks may involve partially ordered actions. Some moves in the task may be performed in any sequence, but others must be done in an exact sequence. A manipulator might be required to pick up a part from a conveyor, position it in five different orientations (for the drilling of five holes), drop the part in a bin, and return to a home position. The holes may be drilled in any order, but the manipulator must go to the bin after the drilling and return to its home position before picking up another part. This operation with its eight points is shown schematically in Fig 3.

The standard TSP solution cannot be applied directly to minimize the task time with the given constraints on moves 8 to 1, and 1 to 2. However, the problem can easily be modified to include these constraints in two different ways. In the first, the C matrix is set up with the individual move times and all the undesired moves are assigned values of infinity. In this case all moves from 8 to any point but 1 are set to infinity. The move times from any point to 1, except 2 from 1, are also set at infinity. A typical C matrix for this problem is given below

$$C=\begin{bmatrix} \infty & 7 & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 4 & 3 & 2 & 9 & 7 & 2 \\ \infty & \infty & \infty & 1 & 8 & 6 & 6 & 5 \\ \infty & \infty & 3 & \infty & 1 & 1 & 9 & 8 \\ \infty & \infty & 8 & 8 & \infty & 6 & 2 & 1 \\ \infty & \infty & 3 & 3 & 1 & \infty & 2 & 2 \\ \infty & \infty & 7 & 5 & 6 & 7 & \infty & 4 \\ 5 & \infty & \infty & \infty & \infty & \infty & \infty & \infty \end{bmatrix}.$$

The optimal tour for this case is

1-2-3-4-6-5-7-8-1

and requires a task time of 25 s. In general, the computation time will be less than that of a normal eight-point system because of the large number of infinities in C.

In the second method for solving this problem, moves 8-1 and 1-2 are treated as a single move in the TSP. Here the C matrix will only have infinities on the main diagonal, and be reduced to a 6 by 6 matrix. This strategy can also be used when the task has continuous motion embedded in point-to-point moves, called semi-continuous operations, as when manipulator must make a number of distinct arc welds. In such a case, each of the welds would be treated as a single point in the TSP. The times in the C matrix to a "weld point" would depend on the direction of the motion along the continuous segment. For this reason an iterative solution may be required in cases which permit motion in either direction [16].

C. Tasks Requiring Tool Changes

Often tasks require a manipulator to change its tools during an operation. Fig. 4 shows a very simple example in which the

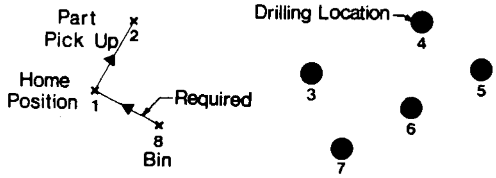


Fig. 3. A partially ordered task: drilling.

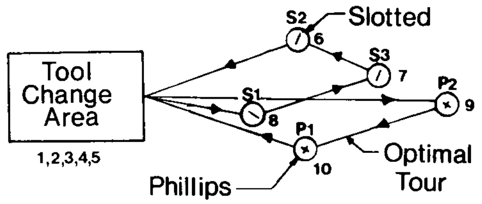


Fig. 4. An example of a task with tool changes.

manipulator is required to tighten a series of screws, some slotted and some with phillips heads. The screws may be tightened in any order, but the manipulator must change tools when switching screw types. The optimal sequence for this operation is not obvious, and again the problem is not in the usual TSP form. However, it can be restructured to make a TSP solution applicable.

Unwanted manipulator moves are prohibited by setting their move times in the C matrix to infinity. Here, the move times between slotted and phillips screws are set to infinity. Furthermore, the manipulator must be able to visit the tool change area before visiting each screw. This is permitted by establishing a set of superimposed tour points in the tool change area, one for each screw. The move times between these points is set equal to zero. With this, the TSP solution is free to select as many tool changes as required to optimize the task. The times between the tool change points and a screw will normally include the time to change tools. The C matrix for the problem shown in Fig. 4 is

$$\begin{bmatrix} \infty & 0 & 0 & 0 & 0 & 29 & 37 & 36 & 32 & 28 \\ 0 & \infty & 0 & 0 & 0 & 29 & 37 & 36 & 32 & 28 \\ 0 & 0 & \infty & 0 & 0 & 29 & 37 & 36 & 32 & 28 \\ 0 & 0 & 0 & \infty & 0 & 29 & 37 & 36 & 32 & 28 \\ 0 & 0 & 0 & 0 & \infty & 29 & 37 & 36 & 32 & 28 \\ 80 & 80 & 80 & 80 & 80 & \infty & 12 & 14 & \infty & \infty \\ 89 & 89 & 89 & 89 & 89 & 4 & \infty & 9 & \infty & \infty \\ 87 & 87 & 87 & 87 & 87 & 11 & 2 & \infty & \infty & \infty \\ 79 & 79 & 79 & 79 & 79 & \infty & \infty & \infty & \infty & 3 \\ 71 & 71 & 71 & 71 & 71 & \infty & \infty & \infty & 6 & \infty \end{bmatrix}.$$

The optimal tour for this case, with a tool change time of 5.0 s, is

1-2-3-4-8-7-6-5-9-10-1

or

[Tool Change]-$S1$-$S3$-$S2$-[Tool Change]-$P2$-$P1$ - $\cdots$.

This sequence takes 22.8 s. Compared to the apparently reasonable, but not optimal, sequence of

[Tool Change]-$S1$-$S2$-$S3$-[Tool Change]-$P1$-$P2$ - $\cdots$

which takes 26.1 s, the optimal operation saves 15 percent. This reduction in time over a number of cycles could substantially increase productivity. The TSP computation time for this solution was 0.6 s. This problem could have also been solved defining the time for a move between different types of screws as the sum of 1) the time for the manipulator to move to the tool change area from the first screw, 2) the tool change time, and 3) the move time to the second screw

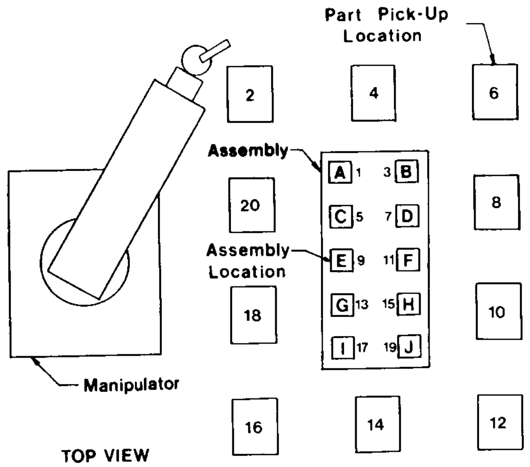


Fig. 5. An assembly station.

from the tool change area. This approach reduces the C matrix size but slightly increases the problem setup effort.

III. Optimal Work Place Design for Robotic Manipulators

One of the most interesting results obtained in this study found that the work places of robotic manipulators could be designed by direct application of the optimal tour building methods to achieve minimum time operations. This approach has proven very effective for setting up assembly operations. Consider the problem shown schematically in Fig. 5. Here 10 different components or parts are to be inserted into specific locations in an assembly by the manipulator. The components are located in 10 different bins which surround the assembly. The question is: "In which bin should each component be placed in order to minimize the assembly time?"

To use the TSP approach to solve this problem, the time for moves which are to be prohibited would first need to be set to infinity. The manipulator should not visit two part bins, labeled with even numbers in Fig. 5, or two insertion points, labeled with odd numbers, in succession. Therefore, in the C matrix all moves between even and odd numbers should be set to infinity. Then, after all permitted move times have been assigned, an optimal tour would be found using the unrestricted TSP solution described above. For this problem C would be a 20 by 20 matrix.

An example of such a solution is contained in [16]. Typically, it takes about 30 s of computation on the PDP 11/44. The optimal work space configuration or bin assignments can be made once the optimal tour is determined. This is accomplished by a kinematic inversion of the partially ordered process described above. In the optimal tour all moves from even to odd points are moves from part bins to assembly or insertion locations. To obtain one simply needs to assign the part required by insertion location i (odd) to the bin location (which will be even) that precedes it in the optimal tour. For example, if the optimal tour for Fig. 5 is

12–1–18–5–4–7–8–11–16–19–20–3–10–17–6–9–14–13–2–15–12

then the parts "H" required by location 15 should be assigned to bin 2.

More complex problems could be handled in a similar fashion. For example, assume an assembly operation which requires that two parts must be placed sequentially, for instance a nut and a washer. This problem could be handled almost exactly like the example described above. There would be one bin for each type of part, but the assembly location to receive the two components would be assigned two odd location numbers. In the optimal tour, the location to receive two components would appear twice because it has two points associated with it. The first part to be assembled should be placed in the first bin location in the tour and the second part in the second location preceeding the assembly point, to insure that the parts at this location are assembled in the proper order.

A related problem is the assembly of the same type of part at many assembly locations. For example, assume that part "B" occupies three places in the assembly. The tour could be optimized with 10 separate part bins, three containing "B" parts being contained in one or two bins. The time for the optimal tours for each of these alternatives would need to be evaluated separately. For one bin containing all three "B" parts, three part locations would need to be superimposed in a particular bin, then the TSP problem would be solved. But the selected bin might not be the optimal one, and therefore, the problem would need to be resolved by locating the "B" parts in each of the possible bins, and by then selecting the shortest tour time. A similar procedure could be applied to cases where there are two bins assigned to "B" parts. While this would appear to require a great deal of effort, in fact it does not. Most of the effort required would be in obtaining the manipulator move times; these are the same for all the cases described above. It would be relatively easy to develop a program to automate the C matrix setup for the TSP solutions. This would be justified for processes where productivity was an important consideration.

IV. Computational Considerations, Clustering in Large-Scale Problems, and Multiple Manipulator Operations

The effort required to determine the interpoint move times can be quite substantial for problems with large numbers of points. The TSP solution computation times also increases for large problems, but this is a less serious problem. If an operation has n points, then $n(n-1)$ move times must be determined for the C matrix. If there are 50 points, 2450 segment times are required. While the TSP solution will require less than 1 min of computation time on the 11/44, the effort to obtain the segment times, even by simulation, would be substantial. An approximate clustering approach was developed to reduce this effort [18]. Clustering consists of 1) grouping neighboring points of a task into clusters, 2) optimizing movements within the cluster, and 3) optimizing moves between clusters. It has been shown that substantial effort could be saved where points formed natural clusters; the task times produced were relatively close to the true global optimal [16], [18].

Finally, multiple manipulator operations were considered in this study. The speed of many industrial operations can be improved by the use of more than one manipulator on a particular task [17]. Examples are where several manipulators make a series of welds on an automotive body, and where they insert many electronic components into a circuit board. This study investigated the question of which manipulators should serve which points to achieve maximum productivity. It was assumed that the multiple manipulators do not interfere with each other, for example, when they work at different stations.

An approximate solution to the problem was obtained by first treating it as if one manipulator were to perform all the operations. Next, this optimal tour was divided into subtours, one for each manipulator, where the time increments for all the subtours are approximately equal [16], [18]. While this approach is not truly optimal, it yielded substantial savings in time over other methods for the cases considered in this study. A true optimal solution to this problem with n points and two manipulators would require

$$\sum_{r=1}^{n} \frac{n!}{r!(n-r)!}$$

TSP solutions. This would present a large computational burden for all but the smallest problems and additional research is needed in this area.

V. Conclusions

This communication shows that it is possible to plan relatively complex manipulator tasks by analytically combining optimal motions for simple moves in an optimal manner. This permits them to be performed in minimum time. The results obtained in this study suggest that substantial increases in system productivity can be achieved by the use of the method proposed in this communication. This method is applicable to industrial tasks where manipulators must go to a number of positions, stopping at each point. This problem is structured into the form of the well-known traveling salesman problem, and it is solved by an exact tour-building type of algorithm based on the branch-and-bound technique. The approach can also be applied to problems in which some ordering of manipulator motions is dictated by a task, problem which generally cannot be treated as a TSP.

It was also shown that nonoptimal move times, obtained from measurement or simulations, can be used with the optimal tour building technique to obtain improved, although suboptimal, task times. Ideally, these should be the optimal times obtained using the methods discussed above for simple motions. However, if these approaches are not used, dynamic simulations and experimental timings may be substituted. Dynamic simulations and experimental timings should result in improved overall task performance over *ad hoc* task planning, but the results will be suboptimal.

Additionally, it was shown that a kinematic inversion approach can be used to structure manipulator tasks and their work environments, so that the tasks can be completed in minimum time. An approximate method for optimizing tasks for more than one manipulator has also been discussed. Exact solutions for this problem do not appear feasible with this approach; additional research is clearly needed in this area.

This algorithm was implemented in Fortran on a relatively small Digital Equipment Corporation PDP-11/44 minicomputer. The optimal motions for the simple moves used to build the overall motion were obtained from a computer-aided design (CAD) software package, called OPTARM, run on a Micro Vax II. Computation time is not critical since the method is clearly an off-line task planning tool. Nonetheless, the computation times required by this approach were quite reasonable, even for these minicomputer systems. Obtaining the individual move times may require substantial effort for very large problems. A method to reduce this effort is suggested in this communication. In any event, the approach is probably best suited for use in a computer-aided design environment with solid modeling and dynamic simulation capabilities.

References

[1] M. E. Kahn and B. Roth, "The near-minimum-time control of open-loop articulated kinematic chains," *J. Dyn. Syst., Meas., Contr.*, vol. 93, no 3, pp. 164-172, Sept. 1971.

[2] J. Y. S. Luh and C. S. Lin, "Optimum path planning for mechanical manipulators," *J. Dyn. Syst., Meas., Contr.*, vol. 102, no. 2, pp. 142-151, June 1982.

[3] J. E. Bobrow, S. Dubowsky and J. S. Gibson, "On the optimized control of robotic manipulators with actuator constraints," in *Proc. 1983 American Control Conf.* (San Francisco, CA, June 22-24, 1983).

[4] ——, "Time-optimal control of robotic manipulators," *Int. J. Robotics Res.*, vol. 4, no. 3, Fall 1985.

[5] S. Dubowsky and Z. Shiller, "Optimal dynamic trajectories for robotic manipulators," in *CISM-IFToMM Symposium on the Theory and Practice of Robots and Manipulators* (Udine, Italy, June 1984). Cambridge MA: MTT Press, 1985.

[6] Z. Shiller and S. Dubowsky "On the optimal control of robotic manipulators with actuator and end-effector constraints," in *Proc. IEEE Int. Conf. on Robotics and Automation* (St. Louis, MO, Mar. 25-28, 1985), pp. 614-620.

[7] V. T. Rajan "Minimum time trajectory planning," in *Proc. IEEE Int. Conf. on Robotics and Automation* (St. Louis, MO, Mar. 25-28, 1985), pp. 759-764.

[8] S. Dubowsky, M. A. Norris, and Z. Shiller, "Time optimal trajectory planning robotic manipulators with obstacle avoidance: A CAD approach," in *Proc. IEEE Int. Conf. on Robotics and Automation* (San Francisco, CA, Apr. 7-10, 1986), pp. 1906-1912.

[9] J. D. C. Little, K. G. Murty, D. W. Sweeney, and D. Karel, "An algorithm for the traveling salesman problem," *Oper. Res.*, vol. 11, pp. 979-989, 1963.

[10] M. Bellmore and G. L. Nemhauser, "The traveling salesman problem: A survey," *Oper. Res.*, vol. 16, pp. 538-558, 1968.

[11] M. Bellmore and J. C. Malone, "Pathology of the traveling salesman subtour elimination algorithms," *Oper. Res.*, vol. 19, pp. 278-307, 1971.

[12] M. Held and R. M. Karp, "The traveling salesman problem and minimum spanning trees: Part II," *Math. Programming*, vol. 1, pp. 6-25, 1971.

[13] V. Klee "Combinatorial optimization: What is the state of the art?" *Math. Oper. Res.*, vol. 5, no. 1, pp. 9-10, Feb. 1980.

[14] A. Tucker, *Applied Combinatorics.* New York, NY: Wiley, 1980, pp. 291-298.

[15] D. K. Smith, *Network Optimization Practice: A Computational Guide.* Market Cross House, West Sussex, England: Ellis Howard Ltd., 1982.

[16] T. D. Blubaugh, "Time-optimal planning of point to point moves for robotic manipulators," M.S. thesis, MIT, Cambirdge, MA, 1985.

[17] D. D. Grossman, R. C. Evans, and P. D. Summers, "The value of multiple independent robot arms," *Robots Computer-Integrated Manuf.*, vol. 2, no. 2, pp. 135-142, Aug. 1985.

[18] S. Dubowsky and T. D. Blubaugh, "Planning time optimal robotic manipulator motions and work places for point to point tasks," in *Proc. 24th IEEE Conf. on Decision and Control* (Ft. Lauderdale, FL, Dec. 11-13, 1985).

Instability Analysis and Robust Adaptive Control of Robotic Manipulators

JOHN S. REED AND PETROS A. IOANNOU

Abstract—**The robustness of adaptive controllers with respect to uncertainty is examined. The uncertainties include bounded input disturbances, unknown and time-varying plant parameters, as well as unmodeled dynamics. A simple example shows instability of a recent manipulator control scheme in the presence of bounded disturbances. Then the adaptive laws for updating the controller parameters are modified so that instabilities are counteracted and robustness is guaranteed.**

I. Introduction

The dynamic control of the multi-link robotic manipulator has intrigued many control systems researchers. The system is characterized by many nonlinearities and strong coupling between the individual links of the manipulator chain. Precise trajectory control will be required in many of the envisioned applications of manipulators (e.g., machining, welding, complex assembly). An attractive alternative to conventional fixed controllers, which may exhibit undesirable overshoot behavior when the plant parameters are unknown or are changed, is adaptive control. By adding provisions to adapt to the dynamic operating environment (e.g., load/tool changes, workpiece variations) as well as to simply identify other unknown and possibly time-varying parameters of the robotic plant on-line, better performance and productivity might be achieved.

Manuscript received November 23, 1987; revised August 22, 1988. This work was supported by the National Science Foundation under Grant DMC-8452002 with matching funds from General Motors Corporation. The work of J. S. Reed was supported under a Hughes Aircraft Doctoral Fellowship.

J. S. Reed is with Hughes Aircraft Company, P. O. Box 92426 R31/D222, Los Angeles, CA 90009.

P. A. Ioannou is with the Department of Electrical Engineering—Systems, University of Southern California, Los Angeles, CA 90089-0781.

IEEE Log Number 882537.