Proceedings of the
1999 ASME Design Engineering Technical Conferences
September 12-15, 1999, Las Vegas, Nevada

DETC99/DAC-8649

MOTION PLANNING FOR MULTI-ROBOT ASSEMBLY SYSTEMS

M. Bonert L.H. Shu B. Benhabib*

*Computer Integrated Manufacturing Laboratory, Department of Mechanical and Industrial Engineering, University of Toronto, 5 King's College Road, Toronto, ON, Canada, M5S 3G8 *email: beno@mie.utoronto.ca*

Keywords: assembly, multi-robot, genetic algorithms, motion planning, electronic component placement

ABSTRACT

This paper addresses a multi-robot optimal assembly planning problem which in essence is an augmented Travelling Salesperson Problem (TSP+). In our TSP+, both the "salesperson" (a robot with a tool) as well as the "cities" (another robot with a workpiece) move. Namely, in addition to the sequencing of tasks, further planning is required to choose where the "salesperson" should rendezvous with each "city". The use of a genetic algorithm (GA) is chosen as the search engine for the solution of the multi-robot TSP+ optimization. As an example industrial application the optimization of the electronic-component placement process is addressed. In the most generalized component-placement system configuration, the placement robots meet the component delivery systems (CDSs) at optimal rendezvous locations for the pick-up of components and subsequently meet the printed circuit board (on the mobile XY-table) at optimal rendezvous locations for their placement. In addition to the solution of the component-placement sequencing problem and the rendezvous-point planning problem, the collision-avoidance issue is also addressed.

1. INTRODUCTION

Autonomous robotic systems are increasingly utilized in industrial environments, requiring the development of modelling methodologies and tools to optimize their operational efficiency. In this context, the classical assembly-planning optimization problem has been extensively studied, with numerous recent attempts at applying the earlier research results to robotic-based assembly, [e.g., 1].

This paper presents a novel point-to-point (PTP) motion-planning technique for multiple coordinated assembly robots, which can be modelled as a TSP, using Genetic Algorithms (GAs) [2]. As an example industrial application, the optimization of an electronic component placement process is utilized [e.g., 3, 4].

The Travelling Salesperson Problem: Many variations of this combinatorial problem have been addressed in the literature, including the Asymmetric, Symmetric, Euclidean, Chebyshev, Prize Collecting and Time-dependent TSP variations [e.g., 5, 6, 7, 8, 9, and 10]. Leu et al. [11] solve the electronic-component placement optimization problem using genetic algorithms. Three types of assembly machines were modelled: (1) A single-robot pick-and-place problem with fixed feeders and a fixed Printed Circuit Board (PCB), (2) Problem 1 extended to have feeder component assignment optimization, and (3) A multi-head fixed placement turret with a moving XY-table and feeder component optimization.

The Augmented Travelling Salesperson Problem (TSP+): In contrast to the single-robot TSPs, where the primary objective is to find the best sequence for N tasks, for multi-coordinated-robot problems, one must also solve the "rendezvous-point" planning problem. In such augmented TSP, (TSP+), the "salesperson" (one robot) as well as the "cities" (PCB placement locations that are moved around by another robot) have motion capability. Namely, further planning is required to choose where the "salesperson" should rendezvous with the "city". This minimum-time optimization problem is further complicated when two placement robots are used concurrently in a task-sharing mode. As one of the few research projects on point-to-point (PTP) TSP+, Cao et al. [12,13] address the issue of inspection-task-sequence planning for two coordinated robots using the simulated-annealing technique. A series of locations on a sphere are inspected by the robot pair. As expected, they showed that the cooperative robot configuration,

where both robots moved, was faster than when only one robot moved and the other acted as a fixture.

Augmented Travelling Salesperson Problem with Multiple Robots: If an additional robot is introduced into the TSP+, which continuously shares its workspace with the other placement robot, it would be necessary to directly address the collision avoidance problem. Such multiple-robot collision avoidance problems have been addressed in the literature in two primary ways: (1) Collision avoidance through path planning, and (2) Collision avoidance through scheduling (or time delays), [14, 15, 16, 17]. For example, Lee and Lee [18] propose solving the collision-avoidance problem by speed changes, which introduce a time delay in one of the two robots. Each robot is represented by a single sphere at the wrist, and straight-line motion is assumed. For paths that are close together, collision maps of time versus distance travelled along the robot path are generated.

2. PROBLEM DEFINITION & SOLUTION APPROACH

In electronic assembly, components must be placed onto the PCB in a time-efficient manner. The first task is the configuration of the PCB, where component locations are determined subject to constraints and objectives. In this research, it is assumed that this task has already been carried out. It is also assumed that the component-placement machine picks and places one component at a time. Although various other placement strategies exist, and are further detailed in the literature, [e.g., 7], the objective of this paper is the investigation of the fundamental TSP+ problem as it applies to electronic component placement.

2.1. Problem Definition

(i) Set-Up Geometry

Figure 1 shows the most generalized physical set up of the two-robot-placement machine modelled. The system comprises five main sub-systems: two X-Y gantry robots for component pick-and-place operations; a numerically controlled X-Y table, on which the PCB is located; and, two single-dof multiple-component delivery systems, with controllable motions in the Y direction.

The two gantry robots share a common workspace, which includes the workspace of their respective component-delivery systems and the workspace of the X-Y table. Each CDS is accessed by only the robot assigned to it. The two robots are not allowed to crossover each other.

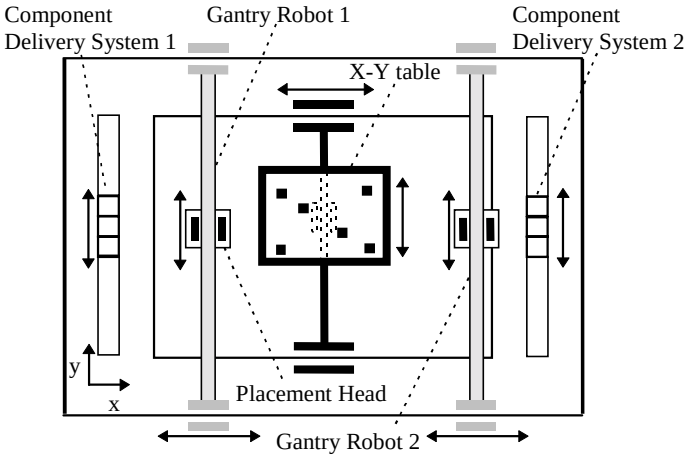

Figure 1: The Two-Robot Electronic-Component Placement Machine Configuration.

(ii) Problem Structure

For the two-robot configuration in Figure 1, there exist five sub-problems: Assigning components to robots; Determining the placement sequence; Finding the rendezvous locations; Coordinating robot movements; and, Planning the device paths.

Although components are placed one at a time, according to an overall sequence, each component is placed by only one of the two robots. This requires each robot to be assigned the components it is responsible for placing. Since components are assumed to be placed sequentially, the placement sequence is identified as a variable. The rendezvous-planning problem is the process of determining the meeting positions of the placement robot and the CDSs, and the meeting positions of the placement robot and the mobile X-Y table. When the combined dof of the two moving sub-systems is above the minimum needed, an infinite number of possible rendezvous-location solutions exist for every potential pick or placement exchange between two moving devices. Therefore, for a given sequence of picks and placements, a corresponding set of optimal rendezvous locations must be determined. Once the rendezvous positions have been determined, since there are two gantry robots sharing the workspace, it is necessary to coordinate the actions of the two robots to prevent collisions.

The final problem is the robot-path-planning problem. For a given point-to-point (PTP) (rendezvous) motion, the fastest robot path is normally determined using the robot dynamics. In our case, for a given multi-component placement sequence, the optimality of a potential set of rendezvous locations can be determined by measuring the overall motion time. To achieve optimal results, individual robot paths between these rendezvous locations must also be optimized. This robot path sub-optimization problem is not addressed in this paper since it has been extensively addressed by the robotics research community [e.g., 19]. Herein, it is simply assumed that

minimum robot-motion time can be achieved by minimizing the Cartesian distance travelled by the individual devices.

(iii) The Assembly Cycle

In order to calculate placement cycle times, the overall board-population assembly problem is divided into individual cycles. Figure 2 shows the process of a generic pick-and-place cycle. The robot starts this cycle at its previous placement location and the CDS starts at its previous pick location. The CDS moves to the current pick location, path # (1), while the robot simultaneously moves there as well to rendezvous with the CDS, path # (2). The robot then picks the component from the CDS. While (1) and (2) are happening, the X-Y table moves to the current placement location, path # (3), where it meets with the robot arriving from the current pick location, path # (4). While (3) and (4) are happening, the CDS is allowed to move, without waiting, to the next pick location. The robot then places the component and the cycle repeats itself for the next component in the placement sequence (with either the same robot or the second robot).

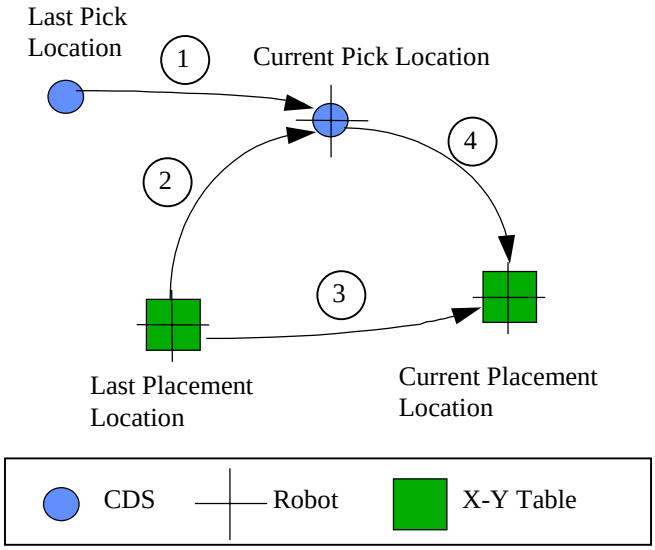

Figure 2: Illustration of a Single Cycle.

2.2. Proposed Solution Approach

The goal of our research was the development of a methodology to optimize the assembly process outlined above. Rather than optimizing one parameter at a time, a method that allows us to optimize all parameters simultaneously is chosen. This is advantageous since many of the parameters are interdependent and the modification of one problem parameter affects another. Use of a GA was chosen as the solution approach for the problem at hand. The parameters for the problems are encoded into genomes (GA data strings). The fitness (objective function value) is used to determine which genomes are chosen for reproduction. The genetic operators, then, generate new offspring, which are evaluated, and the entire process is repeated until the best genome is determined.

The solution strategy must consider collision avoidance between the two placement robots. Since the parameters of sequencing and device positions during pick-and-place operations affect both the collision avoidance problem and the performance of a given placement strategy, it is advantageous to check both simultaneously when searching for an optimum solution, hence their integration into a single objective function in this research.

To encode the two-robot parameters, a compound genome consisting of a series of sub-genomes is proposed: a sequencing sub-genome, two robot-assignment sub-genomes, and a rendezvous-point planning sub-genome.

3. THE TWO-PLACEMENT-ROBOT PROBLEM

The goal of the two-placement-robot TSP+ problem is to minimize the total assembly time. The cycle time is used directly as the basis for the fitness score of a given GA genome. This solution is an extension of a similar single-robot problem detailed in [20]. For clarity purposes, the collision avoidance issue will be addressed, only after the sequencing and rendezvous-point-planning problems have been discussed.

3.1. Solution without Collision Avoidance Considerations

3.1.1. The Objective Function

The objective function evaluates and assigns a fitness value to a genome. Assembly time is used directly as the basis for the fitness score. A genome for the two-robot case, with no collision-avoidance consideration consists of four parts: The sequencing sub-genome, two robot-assignment sub-genomes, and the rendezvous-point sub-genome.

3.1.2. Calculating Cycle Time

For a potential placement sequence, the cycle time per component can be calculated based on the selected rendezvous locations between the devices and their initial locations. In order to convert this collection of positions into time, one needs to know the trajectory of each device from one point to the next. For simplicity, in this paper, it is assumed that all devices move in straight lines between two points at devices' maximum allowable speeds. (Due to the modularity of the proposed solution, one may utilize one of the many robot-trajectory optimization techniques proposed in the literature).

From Figure 3, one can see that there are four motion times that need to be calculated: (i) The robot motion time-to-pick-location, ${}^{pk}_{r}t_i$, (ii) The CDS-motion-time to pick location, ${}_{d}t_i$, (iii) The robot motion time-to-placement-location, ${}^{pl}_{r}t_i$, and, (iv) The X-Y table motion time-to-placement location, ${}_{t}t_i$.

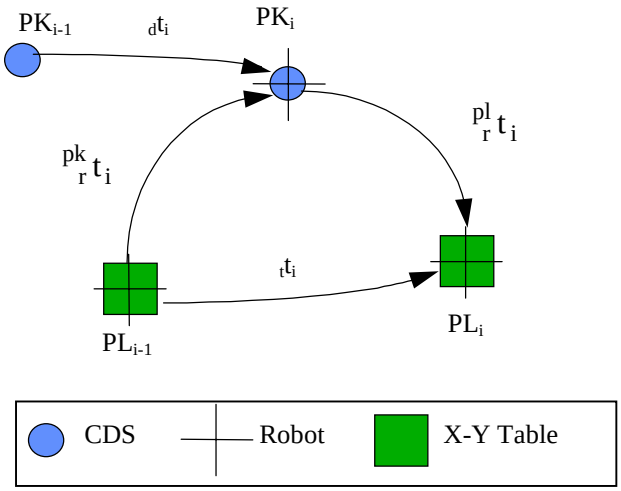

Figure 3: Illustration of Cyclic Device Motion Times.

The overall assembly time for a complete population of a PCB, t, is calculated herein as follows:

$$t = \sum_{i=1}^{N} C_i \,, \qquad (1)$$

where C_i is the time it takes to complete cycle i and N is the number of components on the PCB. One can note that the cycle C_i can be divided into two parts: (i) The time before the end of the component pick operation, and (ii) the time after the component pick operation.

3.1.3. Calculating Motion Times

The cycle time C_i for the two-robot system is defined by:

$$C_i = \max[{}_{r}t_i, {}_{t}t_i] + {}^{pl}c_i \,, \qquad (2)$$

where the robot time, $_{r}t_i$, is the time the robot under consideration executes all tasks required and moves from the last placement location to be ready to place the next component at the current placement location. The X-Y table time, $_{t}t_i$, is the time the table takes to move from the last placement location to the current placement location. The robot cycle time, $_{r}t_i$, can be divided into: (i) the time before the pick operation, and (ii) the time after and including the pick operation.

(i) The robot time before the pick operation

The completion of the first part of the robot cycle time, ${}_{r}t_i^{1st}$, depends on both the CDS' motion and the robot's motion to the pick location. Therefore, the first part of the robot cycle time is the maximum of the time the robot takes to reach the pick location and the time the CDS takes to reach it. The former is the robot motion time to pick location, $^{pk}_{r}t_i$, minus the robot off time, $^{off}_{r}t_i$. The latter is the CDS motion time to pick location, $_{d}t_i$, minus the CDS off time, $^{off}_{d}t_i$. Therefore,

$${}_{r}t_i^{1st} = \max[(^{pk}_{r}t_i - {}^{off}_{r}t_i), ({}_{d}t_i - {}^{off}_{d}t_i)] \,. \qquad (3)$$

The second term in Equation (3), $^{off}_{r}t_i$, is the time the current robot has been off since its last placement operation. If the i'th component is placed by the same robot as the (i-1)'th component, then;

$$^{off}_{r}t_i = 0 \,. \qquad (4)$$

Otherwise, it is placed by the other robot and;

$$^{off}_{r}t_i = \sum_{j=k+1}^{i-1} C_j \,, \qquad (5)$$

where the index k represents the last cycle in which the current robot moved.

The fourth term in Equation (3), $^{off}_{d}t_i$, is the time period that the current CDS has not been involved in a pick operation and has had time to move toward its next pick location. In order to calculate $^{off}_{d}t_i$, the total time of the last cycle in which the CDS was picked from, C_k, is taken, and the time the CDS was busy is subtracted from it. The CDS is busy for the first part of the robot cycle time, ${}_{r}t_k^{1st}$, and the component pick time, $^{pk}c_k$. If the i'th component is picked from the same CDS as the (i-1)'th component, then, k=i-1 and:

$$^{off}_{d}t_i = C_{i-1} - \max[(^{pk}_{r}t_{i-1} - {}^{off}_{r}t_{i-1}), ({}_{d}t_{i-1} - {}^{off}_{d}t_{i-1})] - {}^{pk}c_{i-1} \,. \qquad (6)$$

Otherwise, it is picked from the other CDS, where it is also necessary to add all the other cycles that the CDS has been off:

$$^{off}_{d}t_i = \sum_{j=k}^{i-1} C_j - \max[(^{pk}_{r}t_k - {}^{off}_{r}t_k), ({}_{d}t_k - {}^{off}_{d}t_k)] - {}^{pk}c_k \,, \qquad (7)$$

where the index k is the last cycle in which a component was picked from the CDS under consideration.

(ii) The robot time after and including the pick operation

The second part of the robot cycle time, ${}_{r}t_i^{2nd}$, depends on the pick operation time, ${}^{pk}c_i$, and the robot motion time to placement location, ${}^{pl}_{r}t_i$, which occur sequentially,

$$ {}_{r}t_i^{2nd} = {}^{pk}c_i + {}^{pl}_{r}t_i . \quad (8) $$

Adding the first and second parts of the robot cycle time, the overall robot motion time required in Equation (2) is obtained as follows:

$$ {}_{r}t_i = {}_{r}t_i^{1st} + {}_{r}t_i^{2nd} $$
$$ = \max[({}^{pk}_{r}t_i - {}^{off}_{r}t_i),({}_{d}t_i - {}^{off}_{d}t_i)] + {}^{pk}c_i + {}^{pl}_{r}t_i . \quad (9) $$

3.2. Two-Robot Problem with Collision-Avoidance Considerations

As discussed earlier, the two basic collision-avoidance approaches noted in the literature are (i) the path-planning-based approach, and (ii) the time-delay-based approach. The collision-avoidance approach proposed herein adds a safety delay to one of the two robots, delaying its motion until the other robot is no longer on its path. This method may appear to be an approach based on time delays, however, since collision avoidance is integrated into the GA (as described below) the end result is a combination of path changes and time delays.

For a given GA genome, potential collisions are averted by adding a safety delay to one of the two robots. Thus, the time it takes to complete the assembly for that particular configuration becomes longer. (Namely, the fitness of the genome that is based on the assembly time becomes lower). This is where the correction for collision avoidance as part of the GA-optimization evaluation becomes valuable. Since the variables that affect the optimization of the system also determine whether collisions occur, a separate collision-avoidance correction carried out after the path-planning optimization could interfere with the result. Herein, with the collision avoidance integrated into the optimization, as is possible with the GA, it is still possible to ensure that the solution is not only collision free but also optimized.

3.2.1. The Objective Function

The GA objective function evaluates a genome and assigns a fitness value to it. In this section, the objective function is identical to the one presented earlier with an additional term related to collision avoidance.

3.2.2. Calculating Motion Times

The cycle time C_i is calculated using Equation (2). However, the expression for the robot cycle time is modified by one additional safety-delay term, ${}^{sd}_{r}t_i$:

$$ {}_{r}t_i = \max[({}^{pk}_{r}t_i - {}^{off}_{r}t_i),({}_{d}t_i - {}^{off}_{d}t_i)] + {}^{pk}c_i + {}^{pl}_{r}t_i + {}^{sd}_{r}t_i . \quad (10) $$

Before the use of the safety-delay term in Equation (10), one must determine if a collision is possible at all. It is assumed that the robots do not stop anywhere except at the pick and placement locations. Namely, immediately after a placement operation, the current robot starts to move back to its pick location.

To carry out a collision check, the robot state must be modelled in space and time. In order to simplify the collision detection process, it was decided to model the robots as "walls" moving across the workspace, as shown in Figure 4. This modelling simplification reduces the state of the robot (for collision avoidance calculations) to two variables, the robot X-coordinate and a time at which the robot occupies the corresponding position.

In Figure 4, for the placement of a component, Robot 1 moves from CDS 1 to the placement location. Its maximum travel toward the other robot is the X-coordinate of the placement location of component i, ${}_{r1}w_i$. The current robot (Robot 1, in the example) can collide with the other robot (Robot 2) that may still be in the common workspace. Therefore, in determining whether a safety delay is required, one must check all previous cycles, up to the last cycle, in which the current robot moved. Checking these previous cycles is necessary since the cycle sequence determines only when the components are placed, and not when the associated placement robot starts moving. With a large enough robot-off time, a robot in cycle i can start moving during cycle k+1, and reach the placement location (potentially be in the other robot's path) to wait for the X-Y table.

For a collision to occur, the position and time for the two potentially colliding robots have to match. Therefore, the collision check for the current robot (in cycle i) checks all previous cycles until the current robot last moved (cycle k). To check for an overlap, cycles (k+1) to (i-1) are examined to ensure that the current robot placement location in the X axis (${}_{r1}w_i$) does not crossover with any of the other robot's previous placement locations (${}_{r2}w_{(k+1)}$ to ${}_{r2}w_{(i-1)}$). If an overlap occurs in one or more of the previous cycles, further testing to check the time variables is necessary to confirm or disprove a collision.

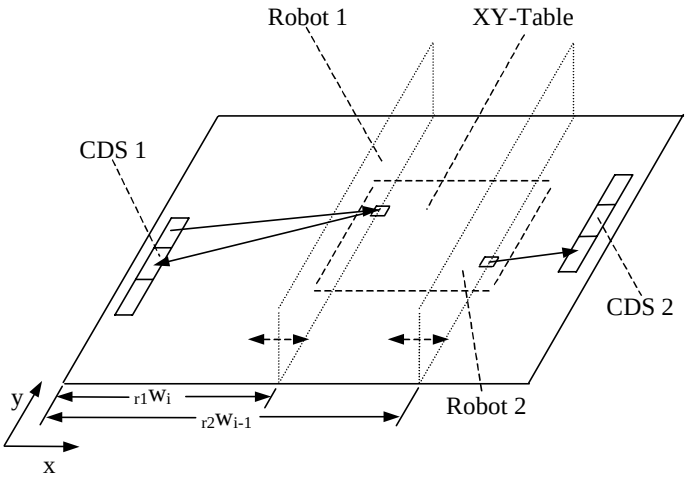

Figure 4: The Robots Modelled as Walls for Collision Avoidance Detection.

In this paper, collision is defined as a situation in which the current robot enters the overlap region before the other robot has left it. In the case where both position and time variables indicate that the robots are in the overlap region at the same time, a preventive strategy must be developed. The approach here is to introduce a delay to the current (cycle's) robot's motion to allow the other robot to leave the overlap region before the current robot enters it.

After all of the cycles, j= (k+1) to (i-1), have been tested for collision, it is possible to calculate the exact amount of time delay required to avoid a collision for a particular cycle j. The delay introduced is the time required by the obstructing robot to clear the overlap area before the current robot enters the overlap area, [21].

3.3. Search Methodology

With the above definition of the objective function and its various input parameters, it is possible to model any multi-robot coordinated system. The methodology adapted in this paper is the simultaneous solution of the multi-level optimization problem using a GA as the search engine. Namely, the optimal genome comprises the best placement sequence of the components as well as the best rendezvous locations between the robotic devices.

Since one of the goals of our work was to explore the effect of the introduction of more dof to an assembly system, our software is reconfigurable to run different set-ups: optimizing the positions of all-fixed devices; optimizing the fixed location of the X-Y table with a mobile CDS configuration; optimizing the fixed locations of the CDSs with a mobile X-Y table configuration; and, an all-devices-moving configuration. The software also allows users to either define the locations of all the components in the bins or have the GA optimize their locations.

3.4. Simulation Examples

The following two configurations were considered for a 6-component board:

(i) All devices moving with user-defined-CDS allocation; Collision-avoidance routine is not employed.

(ii) All devices moving, with user-defined-CDS allocation; Collision-avoidance routine is employed.

For Case (i), the simulation yielded a minimum total time of 1.061 s with a placement sequence of (4, 2, 3, 0, 5, 1). The two robot paths are shown in Figure 5.

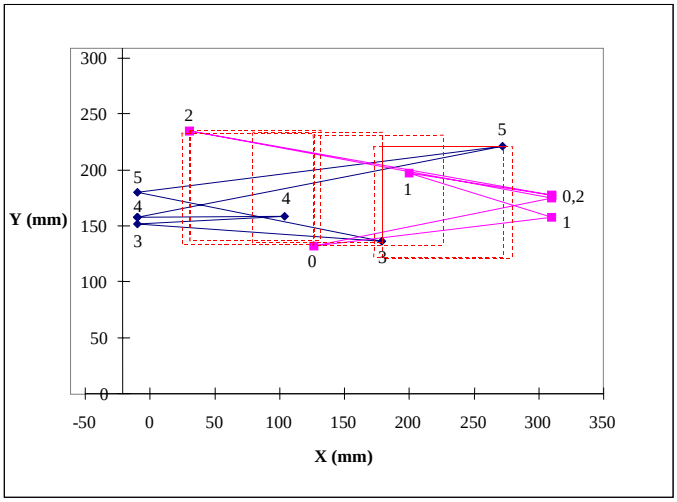

Figure 5: The Robot Paths; Case (i), No Collision Avoidance Routine.

Figure 6 shows the plot of the two robots' positions (X-coordinate) versus time allowing for a quick visual collision identification. On such plots, any location where the line representing Robot 1's movement crosses Robot 2's line indicates a collision event. There are two collision events in the example considered. The first occurs when Robot 2 collides with Robot 1 while trying to place Component 2. The second collision occurs when Robot 2 tries to place Component 1 and collides with Robot 1 placing Component 5.

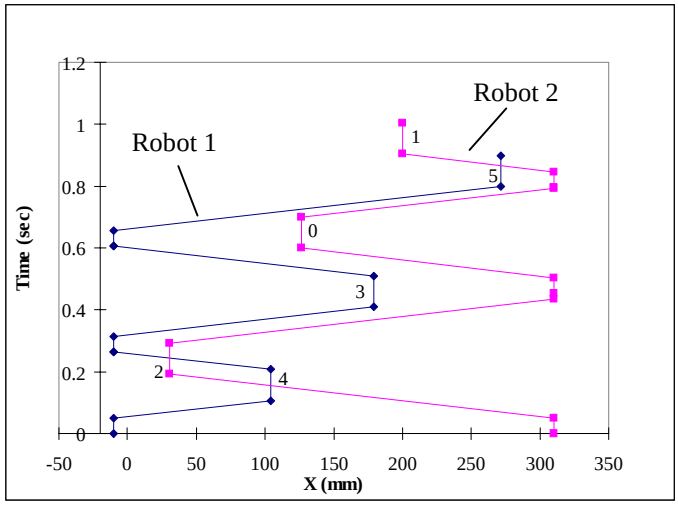

Figure 6: Robot X-Coordinate vs. Time; No Collision-Avoidance Routine.

For Case (ii), where collision-avoidance is considered, the simulation yielded a longer minimum total time of 1.151 s with a different placement sequence of (5, 1, 3, 2, 4, 0). Figure 7 shows the robot paths for Case (ii). Figure 8 shows the robots' positions versus time. In this case, as expected, the robot time lines do not cross. Namely, no collisions occur.

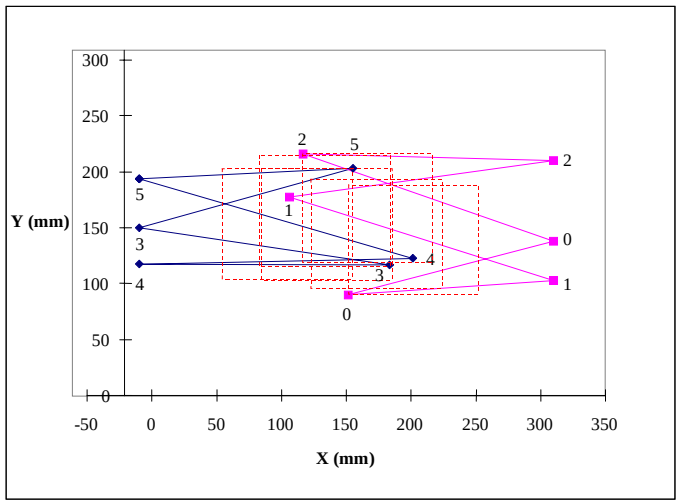

Figure 7: The Robot Paths; Case (ii), With Collision Avoidance Routine.

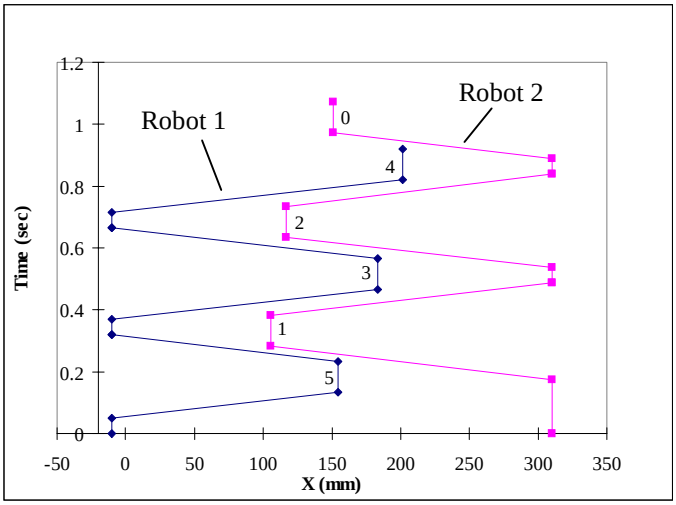

Figure 8: Robot X-Coordinate vs. Time; With Collision-Avoidance Routine.

4. CONCLUSIONS

This paper presented a generalized point-to-point (PTP) motion-planning technique for multiple coordinated assembly robots. The proposed approach minimizes assembly times using Genetic Algorithms (GAs). Specifically, as an example industrial application, the optimization of the electronic component placement process was addressed.

The collision avoidance strategy proposed is a rule-based approach and is directly incorporated into the objective function of the GA. Since the parameters of sequencing and device positions during pick-and-place operations affect both the collision avoidance problem and the performance of a given placement strategy, both are simultaneously optimized.

REFERENCES

1. Drezner, Z. and Nof, S. Y., "On Optimizing Bin Picking and Insertion Plans for Assembly Robots," *IIE Trans.*, 16(3) pp. 262-270, 1984.
2. David E. Goldberg. Genetic Algorithms, in Search, Optimization and Machine Learning. New York: Addison Wesley Publishing Company, 1953.
3. Chang, C., Smith, M. L. and Blair, E. L., "Analysis of Assembly Planning Problem for a Robotized Printed Circuit Board Assembly Center," *Proc. of the 9th International Conference on Production Research*, Cincinnati, Ohio, pp. 472-484, 1987.
4. Francis, R. L., Hamacher, H. W. and Lee, C. Y., "Finding Placement Sequences and Bin Locations for Cartesian Robots," *IIE Trans.*, 26(1), pp. 47-59, 1994.
5. Bozer, Y. A., Schorn, E. C., and Sharp, G. P., "Geometric

Approaches to Solve the Chebyshev Traveling Salesman Problem," *IIE Trans.*, 22(3), pp. 238-254, 1990.
6. Ji, Z., Leu, M. C. and Wong, H., "Application of Linear Assignment Model for Planning of Robotic Printed Circuit Board Assembly," *Trans. of the ASME; Journal of Electronic Packaging*, 114(12), pp. 455-460, 1992.
7. Dubowsky, S. and Blubaugh, T. D., "Planning Time-Optimal Robotic Manipulator Motions and Work Places for Point-to-Point Tasks," *IEEE Trans. on Robotics and Automation*, 5(3), pp. 377-381, 1989.
8. Nelson, K.M. and Wille, L.T., "Comparative Study Of Heuristics For Optimal Printed Circuit Board Assembly," *Southcon Conference Record*, pp. 322-327, 1995.
9. Naft, J. "NEUROPT: Neurocomputing for Multiobjective Design Optimization for Printed Circuit Board Component Placement," *IJCNN: International Joint Conference on Neural Networks*, New York, NY, Vol. 1, pp. 503-506, 1989.
10. Huang, Y., Srihari, K., Adriance, J. and Westby, G., "A Solution Methodology for the Multiple Batch Surface Mount PCB Placement Sequence Problem," *Transactions of the ASME; Journal of Electronic Packaging*, 116(4), pp. 282-289, 1994.
11. Leu, M. C., Wong, H. and Ji, Z., "Planning of Component Placement/Insertion Sequence and Feeder Setup in PCB Assembly Using Genetic Algorithm," *Trans. of the ASME; Journal of Electronic Packaging*, 115(4), pp. 424-432, 1993.
12. Cao, B., Dodds, G. I. and Irwin, G. W., "Time-Suboptimal Inspection Task Sequence Planning for Two Cooperative Robot Arms Using Mixed Optimization Algorithms," *Proc. of the IEEE International Conf. on Robotics and Automation*, Albuquerque, New Mexico, pp. 2103-2108, 1997.
13. Cao, B., Dodds, G. I. and Irwin, G. W., "A Practical Approach to Near Time-Optimal Inspection-Task-Sequence Planning for Two Cooperative Industrial Robot Arms," *International Journal of Robotics Research*, 17(8), pp. 858-867, 1998.
14. Baba, N. and Kubota, N., "Collision Avoidance Planning of a Robot Manipulator by Using Genetic Algorithm. A Consideration for the Problem in Which Moving Obstacles and/or Several Robots are Included in the Workspace," *Proc. of the First IEEE Conference on Evolutionary Computation*, New York, NY, Vol. 2, pp. 714-719, 1994.
15. Zurawski, R. and Phang, S., "Path Planning for Robot Arms Operating in a Common Workspace," *Proc. of the 1992 International Conference on Industrial Electronics, Control, Instrumentation, and Automation; Power Electronics and Motion Control*, New York, NY, Vol. 2, pp. 618-23, 1992.
16. Chang, C., Chung, M. J. and Lee, B. H., "Collision Avoidance of Two General Robot Manipulators by Minimum Delay Time," *IEEE Trans. on Systems, Man and Cybernetics*, 24(3), pp. 517-522, 1994.
17. Baptiste, P., Legeard, B. and Varnier, C., "Hoist Scheduling Problem: An Approach Based on Constraint Logic Programming," *Proc. of the IEEE International Conference on Robotics And Automation*, Los Alamitos, CA, Vol. 2, pp. 1139-1144, 1992.
18. Lee, B. H. and Lee, C. S. G., "Collision-Free Motion Planning of Two Robots," *IEEE Transactions on Systems, Man and Cybernetics*, 17(1), pp. 21-32, 1987.
19. Cao, B. and Dodds, G. I. "Implementation of Near-Time-Optimal Inspection Task Sequence Planning for Industrial Robot Arms," Proceedings of the 4th International Workshop on Advanced Motion Control, Mie, Japan, pp. 693-698, 1996.
20. Bonert, M., Truillet, F., Shu, L. H, Sela, M.N, Benhabib, B. "Optimal Motion Planning of Robotic Systems Performing Point-to-Point Electronic-Assembly Operations," 2nd World Automation Congress, ISORA, Alaska, pp. 22.1-22.6, 1998.
21. Bonert, M., "Motion Planning for Multi-Robot Assembly Systems," M.A.Sc. Thesis, Department of Mechanical and Industrial Engineering, University of Toronto, January 1999.