Motion Planning of Robot Arm with Rotating Table for a Multiple-Goal Task

Yanjiang Huang, Hao Ding, and Xianmin Zhang(✉)

Guangdong Provincial Key Laboratory of Precision Equipment and Manufacturing Technology, South China University of Technology, Guangzhou, China
xmzhang@scut.edu.cn

Abstract. Motion planning of a manipulator system is important when using it to complete a task. In this study, we consider a manipulator system with a robot arm and a rotating table for a multiple-goal task. We first assign the goals around the object into several clusters based on the face in the geometric shape of the object. And then search for the shortest path for the goals in a cluster by coordinating the motion of robot arm and the motion of rotating table based on the particle swarm optimization (PSO). Finally connect the paths found in each cluster. The collisions between the components of the manipulator system is taken into account. The proposed method is compared to a method that only uses nearest neighborhood algorithm (NNA) to coordinate the motion of the robot arm and the rotating table. The effectiveness of the proposed method is verified through a simulation with a set of tasks.

Keywords: Motion planning · Manipulator system · Multiple-goal task · Particle swarm optimization

1 Introduction

A manipulator system that consists of a robot arm and a rotating table has been commonly used to complete a multi-goal task, such as spot welding and inspection [1]. In a multi-goal task, the manipulator end-effector has to reach several goals that are defined by the position and orientation in the robot task space. In realizing a multi-goal task by using the manipulator system with a robot arm and a rotating table, it is important to coordinate the motion of the robot arm and the rotating table to increase the productivity. However, motion planning of the robot arm with the rotating table is complex since the manipulator system is redundant and the search space of motion planning is large and nonlinear.

In previous studies, most researchers focused on the motion planning to find the shortest path for the manipulator end-effector to minimize the task completion time. Some researchers focused on the motion planning of a manipulator system for a multi-goal task [2, 3]. They divided the motion planning problem into two sub-problems: the collision-free shortest path and the travelling-salesman problem. The probabilistic roadmap method (PRM) and the sequential quadratic programming (SQP) were used to solve the motion planning problem in [2] and [3], respectively. However, in these

H. Liu et al. (Eds.): ICIRA 2015, Part III, LNAI 9246, pp. 571–580, 2015.
DOI: 10.1007/978-3-319-22873-0_51

studies, the motion of manipulator was assumed to be collision-free, which is infeasible in real applications. PRM is considered as a state-of-the-art algorithm for solving motion planning because it can approximately represents the connectivity of the free configuration. Many researchers focused on the motion planning of robot based on PRM [4, 5]. In other studies, a phase-plane algorithm was used to minimize the trajectory path by considering the manipulator dynamics and constraints [6, 7]. In [8], a hybrid search algorithm that integrates the nearest neighbor algorithm (NNA) and the Dijkstra algorithm (DA) was used to deal with motion planning for a manipulator system with a 6-DOF manipulator and 1-DOF rotational positioning table. Recently, a multi-objective approach was proposed to solve the motion planning for redundant manipulators [9]. A constrained nonlinear programming incorporated into a sequential quadratic programming was proposed to realize the motion planning and control for redundant robots [10]. However, in these previous studies, some algorithms are computational time consuming, and some algorithms are complex.

In this study, we aim to realize the motion planning of a robot arm with a rotating table to minimize the task completion time for a multi-goal task. To realize the motion planning within a low computational time, we first classify the goals into several clusters, then search for the collision-free path for the goals in the same cluster, finally connect the collision-free paths in each cluster to obtain the minimal task completion time for the multi-goal task. The shortest collision-free path for the goals in the same cluster can be solved by many heuristics algorithms, which can be classified to single solution search algorithms (e.g., Simulated annealing, Tabu search), and population-based search algorithms (e.g., Particle swarm optimization, Genetic algorithm). The population-based algorithms concern a population of solutions at a time, which can overcome the drawback of single solution search algorithm by avoiding the local optimal. In this study, we propose the particle swarm optimization (PSO) to search for the shortest collision-free path for the goals in a cluster, because PSO can search for a good solution with fewer computational time [11, 12].

The problem is formulated in Section 2. The proposed method is described in Section 3. The simulation and discussion are presented in Section 4, and the conclusion is presented in Section 5.

2 Problem Formulation

This section describes the problem solved in this study, including the assumption, input parameters, objective function, constraints and design variables. As shown in Fig. 1, the robot manipulator system consists of a robot arm and a rotating table. The object is located on the rotating table. Serval goals are around the object. The end-effector of the manipulator has to reach the goals to perform some actions.

To solve the motion planning for the robot manipulator system with a robot arm and a rotating table, we maintain the following assumptions:

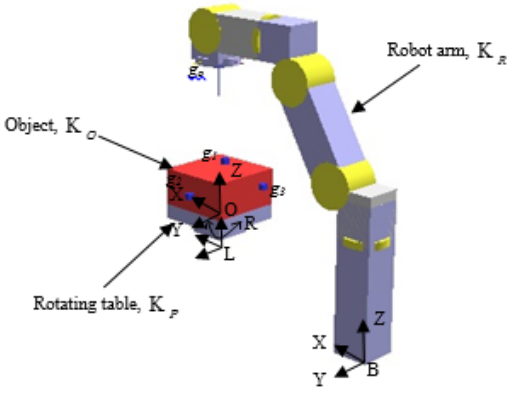

Fig. 1. A robot manipulator system with a robot arm and a rotating table. B is the base position of robot arm, L and O is base of positioning table and object that takes B as reference coordinate frame. R refers to rotational direction.

- The velocity of a robot joint and rotating table is constant when executing a task.
- The shape and size of an object, distribution of goals and goal order are known.
- The distance between the robot base and the rotating table base is known.
- The rotating table can rotate in $[-\pi, \pi]$.
- The specification of goals can be represented by position and orientation. Here, the orientation is the orientation of the end-effector required to perform some action.
- The object can be modelled as a rectangular box, and the goals around the object are outside the box.

The specifications of manipulator system (e.g., D-H parameters of robot arm, size of rotating table) and the specifications of the task (e.g., number of goals, position of goals) are set as input parameters.

The objective of the motion planning is to minimize the task completion time, which can be formulated as follows:

$$\text{Minimize} \quad T = \sum_{i=1}^{m} f(g_{i-1}, g_i) \tag{1}$$

$$f(g_{i-1}, g_i) = \max\left(_{l\in\{1,\ldots,6\}}\left\{\left|\theta_{i-1}^{l} - \theta_i^{l}\right| / V^{l,\max}\right\}, \left|\theta_{i-1}^{r} - \theta_i^{r}\right| / V^{r,\max}\right) \tag{2}$$

where, T is the task completion time that the end-effector of the manipulator from the initial position, passing through each goal. m is the number of goals that the end-effector has to pass through. $f(g_{i-1}, g_i)$ is the motion time of the robot joints and the rotating table from goal g_{i-1} to g_i. g_0 is the initial position. θ_i^l is the angle of robot joint l at goal g_i. $V^{l,\max}$ is the maximal speed of joint l. θ_i^r is the angle of the rotating table at goal g_i. $V^{r,\max}$ is the maximal speed of the rotating table.

The motion planning of the manipulator system can be considered as an optimization problem. The constraints for solving the problem are set as collision-free constraint. The collision-free constraint requires the end-effector reach to each goal without collision among the components of manipulator system (e.g., robot arm and rotating table).

The design variables are set as kinestate of manipulator system $q_i = \left(\theta_i^1 .... \theta_i^6, \theta_i^r\right)$, here, $\left(\theta_i^1 .... \theta_i^6\right)$ are the robot arm joint angles, θ_i^r is the joint angle of the rotating table.

3 Proposed Method

In the manipulator system with a robot arm and a rotating table, the travelling distance of the robot arm depends on the joint space of the robot arm and the rotating table. In this study, the configuration of manipulator system $q_i = \left(\theta_i^1 .... \theta_i^6, \theta_i^r\right)$ at i^{th} goal depends on the angle of rotating table θ_i^r. When the angle of rotating table is determined, the configuration of robot arm can be obtained based on the inverse kinematics (IK). Therefore, we can coordinate the motion of the rotating table to find the shortest collision-free path to pass through each goal. It is difficult to obtain the shortest collision-free path, because the angle of rotating table can be any value from -π to π at each goal, which leads to a large and nonlinear joint space of robot arm. To realize the motion planning for the manipulator system within a reasonable computational time, we first classify the goals into several clusters, and then search for the collision-free path for the goals in the same cluster by coordinating the motion of robot arm and the motion of the rotating table, finally connect the path found in each cluster. The collision detection is taken into account in this study. Because the collision is easy to occur when the rotating table is in some angles, we limit the angle of the rotating table in some range. It is reasonable to do this by considering the computational time cost. To simplify the problem, the goal order and cluster order are known in advanced. The details of goal clustering, motion coordination, and collision detection are described later in this section.

1) Goal Clustering

We assign the goals into several clusters based on their topological location. Each cluster is determined by the face in the geometric shape of the object. Every goal is associated to a cluster $\{C_1 ... C_5\}$, as shown in Fig. 2. It is reasonable to assign the goals to several clusters and search for the shortest collision-free path in a cluster, because the number of dimension can be reduced by using this method. Because the goals in the same cluster are near, the motion time of robot arm and rotating table from a goal to another in the same cluster may be shorter than that moving from a goal to another in different clusters. Furthermore, the motion of robot arm between goals in the same cluster has a greater chance to avoid collision than goals from different clusters.

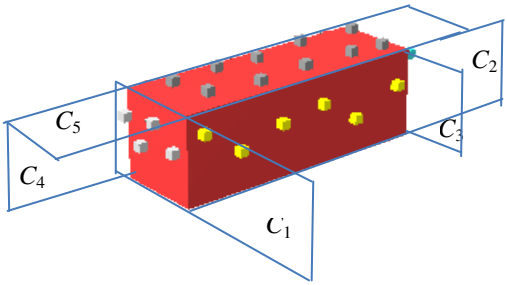

Fig. 2. Cluster of goals. The clusters $\{C_1 ... C_4\}$correspond to the sides of the object while C_5 corresponds to the top of the object. Each goal can only be assigned to one cluster.

2) Motion Coordination

We apply the particle swarm optimization (PSO) to search for the shortest collision-free path in a cluster. PSO is a population-based heuristic algorithm, which is inspired by the movements of a block of birds or fishes when searching for food [13]. The PSO is an evolutionary computation algorithm and in each generation, the velocity and position of a particle are updated by using the following equations:

$$V_i^{h+1} = wV_i^h + c_1 r_1 (pbest_i^h - X_i^h) + c_2 r_2 (gbest^h - X_i^h) \tag{3}$$

$$X_i^{h+1} = X_i^h + V_i^{h+1} \tag{4}$$

Where, $pbest_i^h$ is the best-ever position of a particle; $gbest^h$ is the global best position in swarm; h is generation number; V_i^h is the velocity of the particle i at the h^{th} generation, X_i^h is the position of the particle i at the h^{th} generation, $i = 1,2,..., m$; w is an inertial weight; c_1 and c_2 are cognitive and social scaling parameters; r_1 and r_2 are uniform random numbers between 0 and 1. The PSO algorithm is terminated with a maximal number of generations is reached or the swarm's best position cannot be improved further after some continuous generations.

In the implementation of PSO to search for the configurations of robot arm and rotating table, we set the design parameter as the angle of rotating table at each goal in a cluster. The search space for the goals in a cluster is shown in Fig. 3. The number of possible paths to make the end-effector pass through each goal in a cluster is infinite. It is impractical to search for the shortest collision-free path by an exhaustive method. In this study, we set the number of dimension for a particle is the number of goals in a cluster. Therefore, a particle represents a motion path of the end-effector. The motion time of the robot arm and the rotating table from the first goal to the last goal in a cluster can be calculated based on Equation (2). We first search for a path based on nearest neighborhood algorithm, then limit the search range based on the obtained

path. After that, search for an optimal motion path by using PSO. The process of implementation of PSO is shown in Fig. 4. First, we randomly initialize particles with different positions. The solution with minimal motion time is set as *gbest* . In one iteration, the velocity and position for each particle are updated by Equation (3) and (4). After that, the motion time from the first goal to the last goal in a cluster is calculated by Equation (2). For each particle, the current position is compared to its *pbest* . If the motion time derived by *pbest* is larger than that derived by the current position, then update *pbest* by the current position. Otherwise, retain *pbest* . After all particles are evaluated in one iteration, *gbest* is updated by comparing to the best *pbest* in one iteration. We tune the inertial weight and maximal velocity of the particle based on the Linearly Decreasing $V_{\max}$ method (LDVM) [14] when *gbest* is not improved in the continuous iterations. We terminate the process of motion planning for the goals in a cluster when the maximal number of iterations is reached.

3) Collision Detection

We conduct the collision detection based on oriented bounding boxes (OBBs). We assume that the manipulator system components can be approximately modelled as OBBs, which are defined by the size and orientation. The collision among the manipulator system components can be detected by checking the overlap of the OBBs. There is no collision among the manipulator system components if no collision occurs among the OBBs. Two OBBs are projected onto a potential separating axis. If the projections do not overlap, there is no collision between the two OBBs. If the projections do overlap, the OBBs will be projected onto other potential separating axis and the overlap among projections will be rechecked. 15 potential separating axes can be generated for two OBBs based on the separating axis theorem. A detailed discussion is presented in [15].

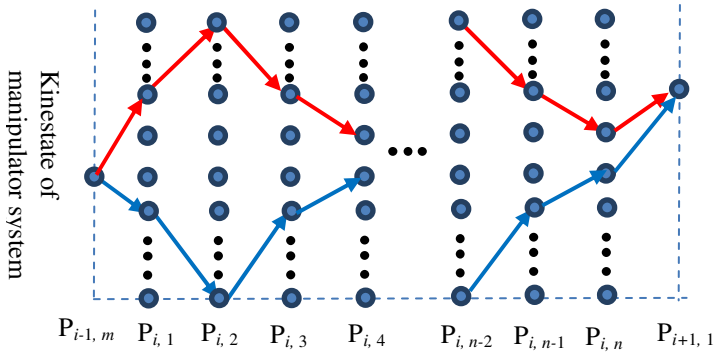

Fig. 3. Search space of motion planning for the goals in a cluster. n is the number of goals in a cluster. $P_{i,\,n}$ is the n^{th} goal in the i^{th} cluster. $P_{i-1,\,m}$ is the last goal in the $(i\text{-}1)^{th}$ cluster. The red arrow and blue arrow mean two different motion paths.

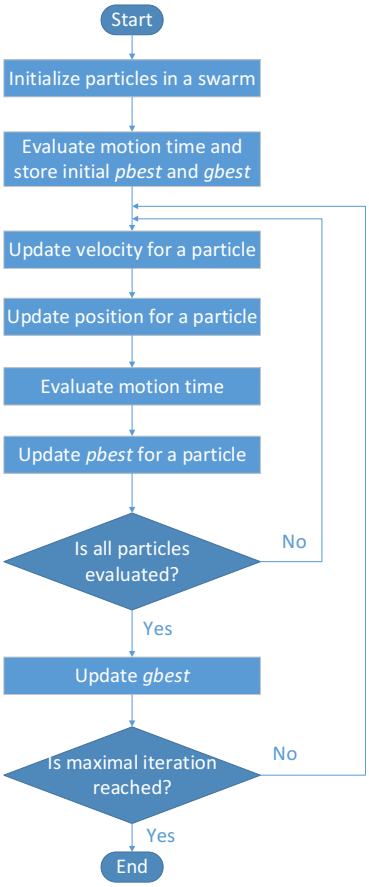

Fig. 4. Implementation of PSO for motion planning for the goals in a cluster

4 Simulation, Result and Discussion

1) Simulation Parameter Setting

We conducted the simulation based on the model of a real robot named VS_6577, which is a robot produced by DENSO WAVE INCORPORATED. The specifications of VS_6577 (e.g., D-H parameters and joint velocity limit) can be found in [16]. The distance between the robot base and the rotating table base is set to 600mm. The table size (length, width, height) is set to (400mm, 400mm, 100mm). The initial position and orientation of the end-effector is set to (500, 0, 600, 0, 180, 0) referred to the robot base.

The object size (length, width, height) is set to (400mm, 400mm, 200mm). We tested three tasks with different number of goals in the simulation. The goals around the object can be assigned into different clusters. The number of goals in each cluster for different tasks is shown in Table 1.

Table 1. Number of goals in each cluster for different tasks

Tasks	C1	C2	C3	C4	C5
Task 1	6	6	6	6	6
Task 2	10	10	10	10	0
Task 3	10	10	10	10	10

We compare the proposed method to the nearest neighborhood algorithm (NNA). We call the proposed method as NNA_PSO, the comparative method as O_NNA. We set the parameters by considering the quality of the solution and the computational time. For PSO, the number of particles is set to 20, the number of generations is set to 100, the parameters w, c1 and c2 are determined by using the method described in [14, 17]. The search step length for NNA is set to 10 degree. The simulation is conducted by using a computer with Intel Pentium(R) 3558U 3.4GHz and 4 GB memory.

2) Simulation Results and Discussion

The simulation results are shown in Fig. 5. The proposed method can perform better than the method that only using NNA to search for the motion path in all the three tasks. The task completion time derived by the proposed method was shortened by 9.38%, 6.83%, 6.18% comparing to the O_NNA in task 1, task 2, and task 3, respectively. In the O_NNA, only using the NNA to search for the shortest path between the two adjacent goals, which is restricted to the local optimal. While in our proposed method, we search for the shortest path for the goals in the same cluster. This is the reason why the proposed method performed better than the O_NNA. When the number of goals around the object increases, the improvement in task completion time decreases. This is because the search space is enlarged when the number of goals in a cluster is increased. It is difficult to obtain the optimal solution in a large search space. If the number of particles or the number of the generations is increased, a better solution may be found. However, more computational time is required when the number of particles or the number of generations is increased. Therefore, the quality of the solution depends on the parameters of the PSO. In this study, we set the parameters for the PSO based on our previous studies by considering the quality of the solution and the computational time.

From these results, we can find that the relationship between the increment of task completion time and the increment of goals is nonlinear. When the goals are located in more faces in the geometric shape of the object, more task completion time is required. This is because more motion time is required for the robot arm moves from a face to another (i.e., the robot arm moves from a cluster to another in this study). Therefore, the optimization of the order for the clusters or the order of the goals in a cluster may be used to shorten the motion time for the robot arm moves from a cluster to another. This can also shorten the task completion time, as described in [1].

Considering the computational time, the proposed method requires longer computational time than that only using NNA, because the proposed method searches for the solution in the global search space. Although the O_NNA requires low computational time, it is easy to fall into local optimal. In a real application, an

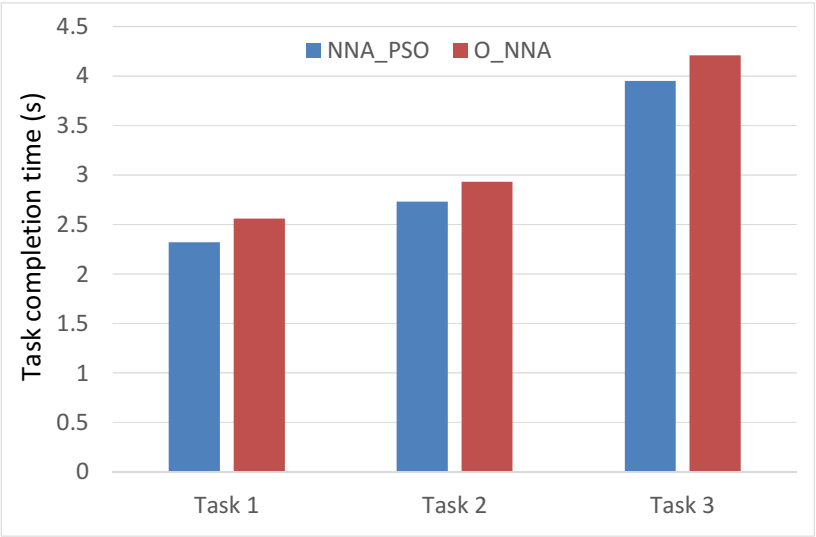

Fig. 5. Task completion time derived by the proposed method and comparative method for different tasks.

appropriate method can be chose based on the constraints in the quality of the solution and the computational time.

5 Conclusion

In this study, we proposed a method that integrates NNA and PSO to realize the motion planning for a robot manipulator system with a robot arm and a rotating table in a multi-goal task. The proposed method is verified to be effectiveness and practical by comparing to the method that only used NNA through a simulation. The task completion time derived by the proposed method is shorted. In the future, the real time motion planning for a robot manipulator system can be taken into account.

Acknowledgment. This work was partially supported by the Fundamental Research Funds for the Central University (Fund No. 20152M004).

References

1. Gueta, L.B., Chiba, R., Arai, T., Ueyama, T., Ota, J.: Practical point-to-point multiple-goal task realization in a robot arm with a rotating table. Advanced Robotics **25**, 717–738 (2011)
2. Saha, M., Roughgarden, T., Latmbe, J.-C.: Planning tours of robotic arms among partitioned goals. International Journal of Robotics Research **25**, 207–224 (2006)

3. Cao, B., Dodds, G.I., Irwin, G.: A practical approach to near time-optimal inspection-task-Sequence planning for two cooperative industrial robot arms. International Journal of Robotics Research **17**, 858–867 (1998)
4. Song, G., Amato, N.M.: A general framework for PRM motion planning. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 4445–4450 (2003)
5. Bohlin, R., Kavraki, L.E.: Path planning using lazy PRM. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 521–528 (2000)
6. Ma, S.: Time optimal control of manipulators with limit heat characteristics of actuators. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 338–343 (1999)
7. Zlajpah, L.: On time optimal path control of manipulators with bounded joint velocities. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 1572–1577 (1996)
8. Gueta, L.B., Chiba, R., Arai, T., Ueyama, T., Ota, J.: Coordinated motion control of a robot arm and a positioning table with arrangement of multiple goals. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 2252–2258 (2008)
9. Marcos, M.G., Machado, J.A.T., Perdicoulis, T.P.A.: A multi-objective approach for the motion planning of redundant manipulators. Applied Soft Computing **12**, 589–599 (2012)
10. Kim, J.H., Joo, C.B.: Optimal motion planning of redundant manipulators with controlled task infeasibility. Mechanism and Machine Theory **64**, 155–174 (2013)
11. Kennedy, J., Eberhart, R.C.: Swarm intelligence. Morgan Kaufmann Publishers, San Francisco (2001)
12. Hassan, R., Cohanim, B., de Weck, O., Venter, G.: A comparison of particle swarm optimization and the genetic algorithm. In: Proceeding of the 1st AIAA Multidisciplinary Design Optimization Specialist Conference, pp. 1–13 (2005)
13. Fourie, P.C., Groenwold, A.A.: The particle swarm optimization algorithm in size and shape optimization. Structural and Multidisciplinary Optimization **23**, 259–267 (2002)
14. Gottschalk, S., Lin, M.C., Manocha, D.: OBB-tree: a hierarchical structure for rapid interference detection. In: Proceeding of ACM Annual Conference on Computer Graphics and Interactive Techniques, pp. 171–180 (1996)
15. Website of DENSO WAVE INCORPORATED. http://www.denso-wave.com/en/robot/product/five-six/vs__Spec.html
16. Huang, Y., Gueta, L.B., Chiba, R., Arai, T., Ueyama, T., Ota, J.: Selection of manipulator system for multiple-goal task by evaluating task completion time and cost with computational time constraints. Advanced Robotics. **27**, 233–245 (2013)