Task constrained motion planning for 7-degree of freedom manipulators with parameterized submanifolds

Qiang Qiu and Qixin Cao
Shanghai Jiao Tong University, Shanghai, China

Abstract

Purpose – This paper aims to use the redundancy of a 7-DOF (degree of freedom) serial manipulator to solve motion planning problems along a given 6D Cartesian tool path, in the presence of geometric constraints, namely, obstacles and joint limits.

Design/methodology/approach – This paper describes an explicit expression of the task submanifolds for a 7-DOF redundant robot, and the submanifolds can be parameterized by two parameters with this explicit expression. Therefore, the global search method can find the feasible path on this parameterized graph.

Findings – The proposed planning algorithm is resolution complete and resolution optimal for 7-DOF manipulators, and the planned path can satisfy task constraint as well as avoiding singularity and collision. The experiments on Motoman SDA robot are reported to show the effectiveness.

Research limitations/implications – This algorithm is still time-consuming, and it can be improved by applying parallel collision detection method or lazy collision detection, adopting new constraints and implementing more effective graph search algorithms.

Originality/value – Compared with other task constrained planning methods, the proposed algorithm archives better performance. This method finds the explicit expression of the two-dimensional task sub-manifolds, so it's resolution complete and resolution optimal.

Keywords Path planning, Redundancy, Autonomous robots

Paper type Research paper

1. Introduction

degree of freedomMany industrial applications are defined as end-effector motions, such as assembling, welding and polishing. In these circumstances, the industrial robots should not only follow the specified end-effector paths but also fulfill some secondary tasks, such as avoiding singularity and collision objects.

degree of freedomdegree of freedomThe redundant robots are built to accomplish these objectives. As shown in Figure 1, Motoman SDA has two 7-DOF arms, which have a similar configuration as human. The human-like configuration is designed for human-level dexterity. In the traditional way, this kind of tasks is realized by human teaching; however, the teaching process of SDA is much more difficult and more time-consuming than 6-DOF robots because of the kinematics complexity.

We want to replace this teaching process with motion planning method and to provide better usability of redundant robots. As the end-effector path can be generated with CAM (computer-aided manufacturing), we focus on finding feasible joint path with given end-effector paths.

2. Related works

If we only consider the end-effector tasks, the joint path between two end-effector waypoints can be simply generated with pseudo-inverse of Jacobian matrix. This method can find the least norm joint path but cannot deal with other secondary tasks.

Interestingly, there is a null space for redundant robots' Jacobian matrix and the self-motion in the null space will not affect the end-effector pose. Many researches have been done on projecting secondary task motions into the null space of the robots' Jacobian matrix. The secondary tasks can be joint limits avoidance (Dubey *et al.*, 1991), singularity avoidance (Yoshikawa, 1984) or collision avoidance (Maciejewski and Klein, 1985; Shen *et al.*, 2015). Besides, Sadeghian *et al.* realized null-space compliance (Sadeghian *et al.*, 2014) and Flacco *et al.* extended it into acceleration level (Flacco *et al.*, 2012).

Similarly, Khatib proposed another control based method, named operational space formulation (OSF) (Khatib, 1987) The OSF can deal with multi-constraints in torque-level control, and Yosuke *et al.* (Kamiya *et al.*, 2013) introduced this method to industrial robot with forward dynamics calculation. These local optimization approaches can be run in real-time but cannot deal with local minimum.

The current issue and full text archive of this journal is available on Emerald Insight at: **www.emeraldinsight.com/0143-991X.htm**

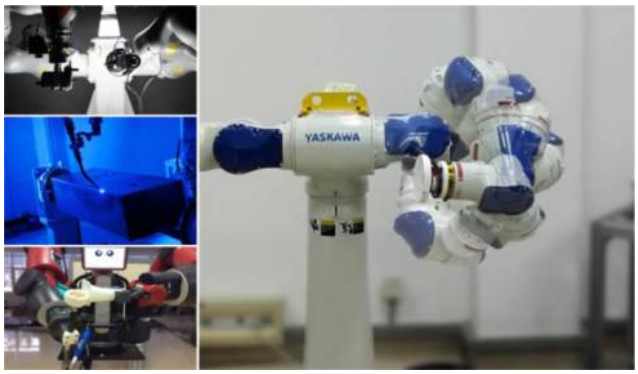
Industrial Robot: An International Journal
45/3 (2018) 363–370

[DOI 10.1108/IR-01-2018-0004]

This work was partially supported by the National Natural Science Foundation of China under grants No. 61673261 and Shanghai Kangqiao Robot Industry Development Joint Research Centre.

Received 13 January 2018
Revised 12 February 2018
5 March 2018
Accepted 12 March 2018

Figure 1 Some industrial applications with specified end-effector paths (left) and the redundancy resolution of Motoman SDA 5F robot (right)



As the control based methods cannot guarantee completeness, some researchers turned to sampling based methods, such as RRT (LaValle, 1998) and PRM (Kavraki *et al.*, 1996), which are probabilistically complete and widely used in general motion planning problem. The basic idea is to project randomly sample configurations to the constrained submanifolds in some fashion. Yakey *et al.* (2001) proposed randomized gradient descent (RGD) method for closed chain mechanism, and Yao *et al.* extended RGD for arbitrary constraints (Yao and Gupta, 2005). However, RGD requires considerable parameter tuning for more general pose constraints.

Besides, Berenson *et al.* (2011) extended the Bi-RRT algorithm to constraint manifolds and proposed CBi-RRT (Berenson *et al.*, 2009). CBi-RRT randomly samples configurations in Euclidean space and use Jacobian pseudo-inverse projection method to project the samples to the task submanifolds. However, owing to the highly non-linear relationship between Euclidean space and task manifolds, the samples are not uniformly distributed on the constraint manifolds. To solve this problem, Kim *et al.* (2016) proposed TB-RRT, and it samples configurations in the tangent bundle of the constraint manifolds.

However, these sampling-based methods may generate jerky or redundant motions owing to the nature of random sampling. These uncertainties of planning result would limit their applications in industrial fields.

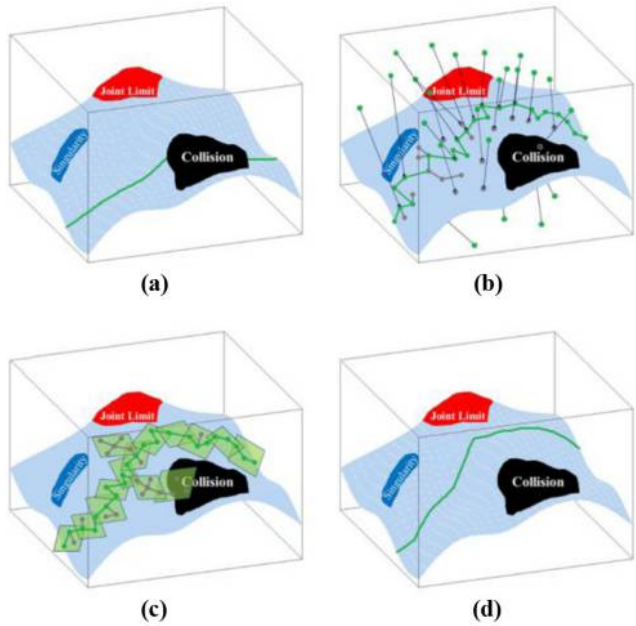
As for a 7-DOF robot with specified end-effector paths, the task submanifolds only have two dimensions. If we can find the explicit expression of the submanifolds, the resolution complete and resolution optimal path can be found by global graph search algorithms. Figure 2 illustrates the difference between these algorithms. This paper proposed a parameterization method, which explicitly describes the submanifolds with two parameters and implemented this method on Motoman SDA robot.

3. Manifolds parameterization

3.1 Overview

The path following task submanifolds for 7-DOF has two dimensions, one dimension is defined by the task; and the other is the self-motion for a specified end-effector pose. In this section, we will introduce a parameterization method for task submanifolds description.

Figure 2 Illustrations of the four motion planning algorithms on constraint manifolds



Notes: (a) Jacobian pseudo-inverse method; (b) CBi-RRT; (c) TB-RRT; (d) our method with parameterized task submanifolds

3.2 Cartesian dimension

For an industrial application, such as robot welding, the CAM can generate a series of Cartesian waypoints (6-DOF end-effector poses) and the robot need to follow these waypoints. The first dimension of the task submanifolds is along the end-effector path, and we should formulate this dimension with a single parameter.

Consider an industrial task defined as a series of waypoints $\{(\boldsymbol{p}_0, \boldsymbol{q}_0), (\boldsymbol{p}_1, \boldsymbol{q}_1), \ldots, (\boldsymbol{p}_i, \boldsymbol{q}_i), (\boldsymbol{p}_{i+1}, \boldsymbol{q}_{i+1}), \ldots, (\boldsymbol{p}_n, \boldsymbol{q}_n)\}$, where $\boldsymbol{p}_i$ is the position of the i^{th} waypoints, and $\boldsymbol{q}_i$ is the quaternion of the i^{th} waypoints.

The basic idea of parameterization is to generate a continuous path via these waypoints in workspace, and then this one-dimensional path can be formulated as a function of a single parameter. In our realization, a simple linear position interpolation and quaternion slerp method are used.

The distance between two waypoints is defined as formula (1):

$$d_k = \|\boldsymbol{p}_{k+1} - \boldsymbol{p}_k\| = \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2 + (z_{k+1} - z_k)^2} \quad (1)$$

And the cumulative distance of the path is defined as follow:

$$D_k = \sum_{i=0}^{k} d_k \quad (2)$$

The parameter s is the distance from the first waypoint along the path, so the Cartesian dimension can be formulated as follow:

$$\begin{aligned} \boldsymbol{p}(s) &= (1-s')\cdot\boldsymbol{p}_i + s'\cdot\boldsymbol{p}_{i+1} \\ \boldsymbol{q}(s) &= \frac{\sin[(1-s')\Omega]}{\sin(\Omega)}\boldsymbol{q}_0 + \frac{\sin[s'\Omega]}{\sin(\Omega)}\boldsymbol{q}_1 \\ s' &= \frac{s-D_k}{d_k}, \text{ and } (D_k \le s < D_{k+1}) \end{aligned} \tag{3}$$

The first Cartesian dimension has been parameterized by a single parameter s, and all task specified waypoints $\boldsymbol{x}$ can be accessed by s, that is:

$$\boldsymbol{x} = \boldsymbol{x}(s) \tag{4}$$

3.3 Self-motion dimension

The second dimension is constrained on the self-motion manifolds, which are the null space of the Jacobian matrix for a fixed end-effector pose. To parameterize the self-motion manifolds, in other word, is to access all feasible inverse kinematics for a given pose via a single parameter.

As the self-motion manifolds are defined as the null space of the Jacobian matrix, the most obvious approach is to find the parameterization method from Jacobian matrix.

First, consider the Jacobian-based inverse kinematics (IK). Jacobian-based IK is a universal solution for serial manipulators, and a feasible IK can be accessed by iteratively minimizing the Cartesian error with formula (5):

$$d\boldsymbol{\theta} = \boldsymbol{J}^{\dagger} d\boldsymbol{x} \tag{5}$$

Here, $\boldsymbol{J}^{\dagger}$ is the pseudo inverse of robot Jacobian matrix.

Only one solution would be returned in a single query, different solutions can be calculated by changing the initial seed configuration. But it's difficult to access all feasible IK solutions with this method.

Second, let's try the null space projection method. As described before, some researchers use the null space projection method to achieve some subtasks, such as singularity avoidance (Yoshikawa, 1985), obstacle avoidance (Nakamura *et al.*, 1987) and torque optimization (Suh and Hollerbach, 1987):

$$d\boldsymbol{\theta} = \boldsymbol{J}^{\dagger} d\boldsymbol{x} + \left(\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J}\right)\boldsymbol{v} \tag{6}$$

As described in formula (6), $(\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J})$ is the null space projection matrix and it can project all vectors $\boldsymbol{v} \in R^7$ into the null space of Jacobian matrix. In other word, different IK solutions can be accessed by continuously changing the vector $\boldsymbol{v}$.

Unfortunately, all the IK solutions found by this method are located on a same self-motion manifold, and there are multiple self-motion manifolds for a specified end-effector pose.

Finally, we find a perfect solution in (Shimizu *et al.*, 2008). Shimizu *et al.* proposed an analytical inverse kinematic solution for 7-DOF redundant manipulators, which can obtain all feasible IK solutions.

They introduced a new parameter called arm angle to increase the dimension of robot pose (6d end pose + arm angle), with this method, all seven joint values are functions of arm angle for specified end pose.

As shown in Figure 3, the Motoman SDA has the similar kinematic configuration as that in (Shimizu *et al.*, 2008), so all feasible IK solutions can be obtained, and all IK solutions on self-motion manifolds can be accessed by a single parameter φ, that is:

$$\boldsymbol{\theta} = \boldsymbol{\theta}(\varphi) \tag{7}$$

Therefore, the whole task submanifolds is parameterized with only two parameters, Cartesian distance s and arm angle φ, and the whole submanifolds can be accessed with these two parameters.

4. Global search

4.1 Overview

In previous section, the task submanifolds is parameterized with two parameters, and all configurations on this submanifolds are accessible. In this section, a graph is constructed on the constraint submanifolds, and graph search method will be used to solve the constrained motion planning problem.

4.2 Cell decomposition

To implement global graph search algorithms, the submanifolds should be decomposed into small cells. With the help of the parameterized submanifolds, the task manifolds can be easily decomposed by parameter discretization.

With the cell decomposition method, we obtain many nodes $\{\boldsymbol{P}_{i,j}\}$ in configuration space and:

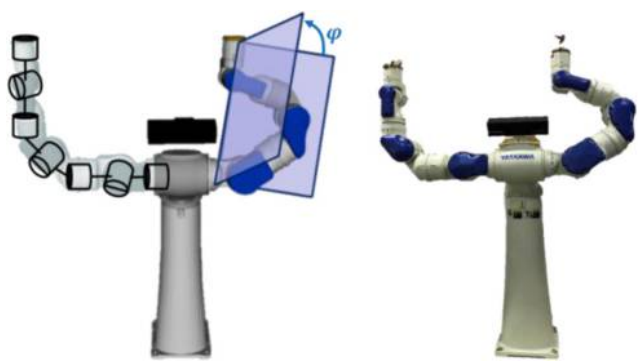
$$\boldsymbol{P}_{i,j} = \boldsymbol{P}(s_i, \varphi_j) \tag{8}$$

And $1 \le i \le m$, $1 \le j \le n$.

Because the cell decomposition method is resolution complete and resolution optimal, a smaller step size in parameter discretization can lead to better planning result.

The submanifolds are explicit parameterized, so the cell decomposition process would be very fast. However, higher resolution would cost more time in graph searching process.

Figure 3 Kinematic configuration of Motoman SDA and the arm angle φ



4.3 Graph connection

The shape of this redundant manipulator's configuration space (C space) is a 7-dimensional torus, so the straight line (shortest path) in C space should be a curve in workspace. If the distance between two waypoints is too large, some error would be introduced into the final end-effector path, we could call it step error.

To improve the trajectory accuracy, the step error should be bounded. Here, we proposed a geometric method to calculate the upper bound of step error for two given points on the task manifolds. What's more, when constructing the searching graph, we connect the point pairs with small step errors (smaller than a human specified threshold).

Let $\boldsymbol{X}$ denote the C space of the redundant manipulator, with local coordinates $\boldsymbol{\theta} \in \mathbb{R}^7$, $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7)^T$. Let $\boldsymbol{M}$ be the task submanifolds of following a specified end-effector paths, and $\boldsymbol{u} \in \mathbb{R}^2$, $\boldsymbol{u} = (s, \varphi)$ denotes the local coordinate for $\boldsymbol{M}$. $\boldsymbol{M}$ is a 2-dimensional surface that are embedded in $\mathbb{R}^7$.

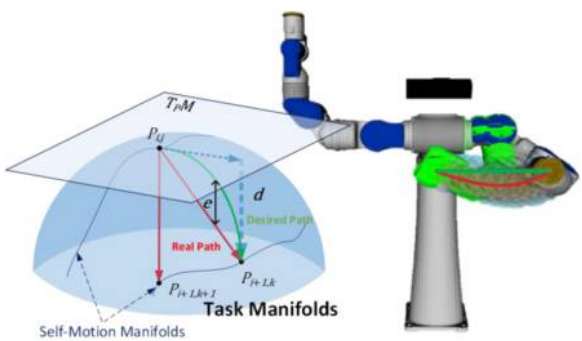
As shown in Figure 4, the blue surface $\boldsymbol{M}$ is the task manifolds embedded in C space $\boldsymbol{X}$, $\boldsymbol{P}_{i,j}$ is an arbitrary point lying on $\boldsymbol{M}$ and m^{th} self-motion manifolds, $T_{\boldsymbol{P}}\boldsymbol{M}$ is the tangent space to $\boldsymbol{M}$ at $\boldsymbol{P}_{i,j}$, $\boldsymbol{P}_{i+1,k}$ is a point in the next self-motion manifolds. The green curve is the desired path on task manifolds, and the red curve is the real path (the shortest path in C space), so e is the step error in C space, and d denotes the upper bound of e:

$$\|e\| \leq d \tag{9}$$

On one hand, all points $\boldsymbol{P}$ on the task constrained manifolds can be represented with local coordinate $\boldsymbol{\theta}$, so all these points can be implicitly characterized by algebraic equality constraints with the form $f(\boldsymbol{\theta}) = 0$. On the other hand, these points can be locally parametrized by $\boldsymbol{u}$ and $\boldsymbol{\theta} = \boldsymbol{\theta}(\boldsymbol{u})$:

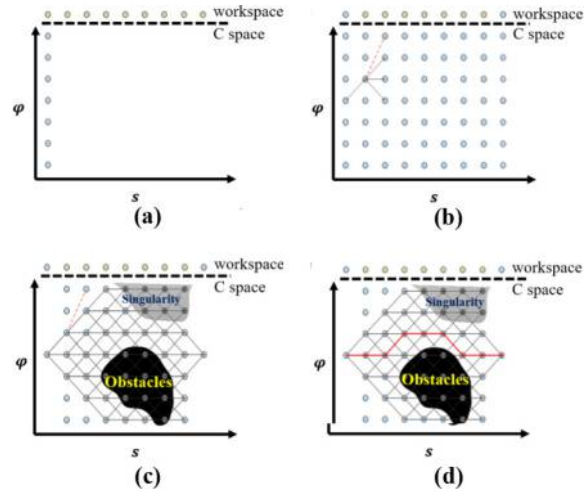
$$\frac{\partial \boldsymbol{\theta}}{\partial u_1} = \frac{\partial \boldsymbol{\theta}}{\partial s} = \frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{x}} \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s} = \boldsymbol{J}^{\dagger} \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s} \tag{10}$$

Figure 4 Illustration of the step error e in C space (left) and real robot (right)



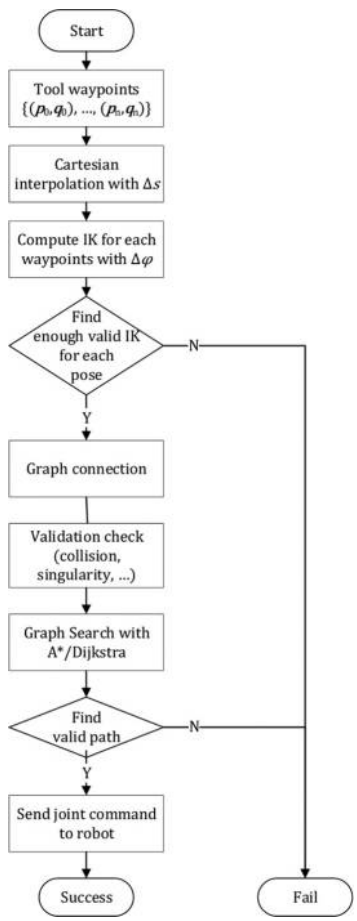
Notes: The blue surface $\boldsymbol{M}$ is the task manifolds, $\boldsymbol{P}_{i,j}$ is an arbitrary point lying on $\boldsymbol{M}$, $\boldsymbol{P}_{i+1,j}$ is a point on the next self-motion manifolds. The green curve is the desired path on task manifolds, and the red curve is the real path (the shortest path in C space), so e denotes the step error in C space

Figure 5 Illustrations of global search



Notes: (a) Cell decomposition on task submanifolds; (b) graph connection with bounding error; (c) validation checking; (d) find valid path with global graph search method

Figure 6 Flowchart of our method



$\boldsymbol{J}^{\dagger}$ is the pseudo-inverse of Jacobian matrix at $\boldsymbol{P}_{i,j}$, $\mathrm{d}\boldsymbol{x}/\mathrm{d}s$ can be calculated from formula (4):

$$\frac{\partial \boldsymbol{\theta}}{\partial u_2} = \frac{\partial \boldsymbol{\theta}}{\partial \varphi} \tag{11}$$

can be obtain from formula (7).

The two vectors (10) and (11) are both tangent to $\boldsymbol{T}_{\boldsymbol{P}}\mathcal{M}$, so the normal vector should be:

$$\boldsymbol{n} = \frac{\left(\frac{\partial \boldsymbol{\theta}}{\partial u_1} \times \frac{\partial \boldsymbol{\theta}}{\partial u_2}\right)}{\left\| \frac{\partial \boldsymbol{\theta}}{\partial u_1} \times \frac{\partial \boldsymbol{\theta}}{\partial u_2} \right\|} \tag{12}$$

Then, as defined in (Kobayashi and Nomizu, 1996), the second fundamental form of the task manifolds is:

$$II = \langle \ddot{\boldsymbol{\theta}}, \boldsymbol{n} \rangle = L\mathrm{d}s^2 + 2M\mathrm{d}s\mathrm{d}\varphi + N\mathrm{d}\varphi^2 \tag{13}$$

and:

$$\begin{aligned} L &= \left\langle \frac{\partial \boldsymbol{\theta}^2}{\partial s \partial s}, \boldsymbol{n} \right\rangle \\ M &= \left\langle \frac{\partial \boldsymbol{\theta}^2}{\partial s \partial \varphi}, \boldsymbol{n} \right\rangle \\ N &= \left\langle \frac{\partial \boldsymbol{\theta}^2}{\partial \varphi \partial \varphi}, \boldsymbol{n} \right\rangle \end{aligned} \tag{14}$$

Finally, the upper bound of the step error is defined as follow:

$$e \le d \approx \frac{1}{2} II \tag{15}$$

Therefore, we connect all point pairs with small step errors, so that the trajectory error would be smaller than the upper bound of the step error.

4.4 Validation checking

The task constrained manifolds are turned into a directed graph with previous processes. In this section, all invalid nodes and their edges should be removed from this graph, such as nodes in collision and nodes near singular configurations.

4.4.1 Collision detection

The Flexible Collision Library (FCL) (Pan *et al.*, 2012) can be used to calculate the minimum distance between robot and the surrounding obstacles. For each node, if its distance is smaller than a human specified threshold, this node and its edges would be removed from the graph.

4.4.2 Singularity avoidance

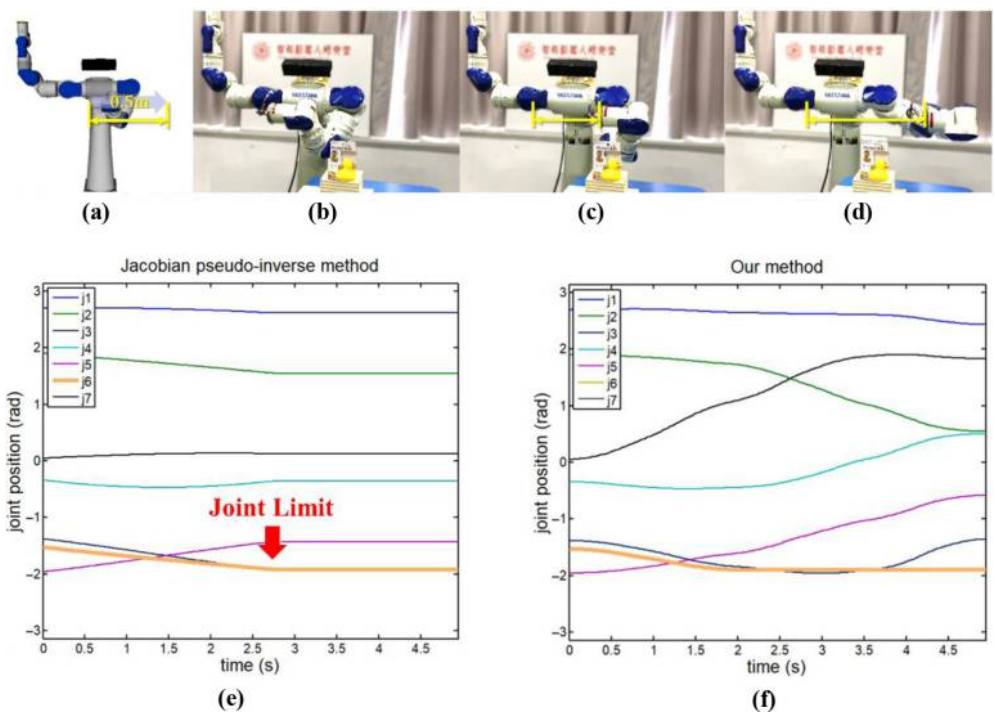
Let $\boldsymbol{J}(\theta)$ denote Jacobian matrix of the manipulator at configuration θ, when $\det(\boldsymbol{J}(\theta)) = 0$, the manipulator is in singular configuration. Yoshikawa (1984) have proposed a measure of manipulability w for serial manipulators:

$$w = \sqrt{\det\left(\boldsymbol{J}(\theta)\boldsymbol{J}^{T}(\theta)\right)} \tag{16}$$

So, to avoid singularity, we just simply remove all nodes with small manipulability and their edges from the graph.

What's more, any other validation functions can be applied to the graph and remove the invalid nodes and edges.

Figure 7 (a) Experiment a: joint limit avoidance; (b) the start configuration; (c) the final configuration of Jacobian pseudo-inverse method; (d) the final configuration of the proposed method; (e) the joint trajectory of the robot with Jacobian pseudo-inverse method; (f) the joint trajectory of the robot with the proposed method



4.5 Global graph search

As shown in Figure 5(d), the task constrained motion planning problem is turned into a simple global search problem on a directed graph, so the shortest path can be found with any graph search algorithms, such as Dijkstra (1959) and A* (Hart *et al.*, 1968). And the flowchart of our method can be found in Figure 6.

5. Experimental results

Three experiments were carried out on Motoman SDA to verify the performance of the proposed method. For the first experiment, the robot should follow a specified end-effector path, and the start configuration was close to joint limits. The second experiment was set up to verify the collision avoidance performance, and the workspace is surrounded by obstacles. To evaluate the performance of our method, we set up the third experiment in a simulation environment, and recorded the planning time, success rate and path error of different methods.

5.1 Joint limits avoidance

As shown in Figure 7(a), the left arm of the Motoman SDA must follow a straight line, and the start tool pose in robot's base frame is 0.6, –0.1, 0.9, 0, $-\pi/2$ and 0, and the end tool pose is 0.6, 0.4, 0.9, 0, $-\pi/2$ and 0. Both Jacobian pseudo-inverse method and the proposed method were used. This example is meant to demonstrate that our method can handle joint limits while still following a given end-effector path.

The experiment results can be found in Figure 7. The Jacobian pseudo-inverse method can find the least norm joint path, but it cannot deal with joint limits; the sixth joint meets its limit at about 53 per cent of the end-effector path.

Our method is a global graph search method, and all configurations out of joint limits are removed from the graph, so it would not drive our robot out of joint limits. As shown in Figure 7, Motoman SDA can deal with joint limits well and finish the specified end-effector path.

5.2 Collision avoidance

In this problem, the robot is surrounded by obstacles, and the given end-effector path is shown in Figures 8(a)-(b). This experiment is similar to the one in (Kucuk, 2016), and the positions of the obstacles are hard-coded, and FCL is used to compute the distance between robot and the surrounding obstacles. Three methods are implemented: simple Jacobian pseudo-inverse method, null space projection method from (Shen *et al.*, 2015) and our method.

The simple Jacobian pseudo-inverse method always follows the least norm joint path, so it cannot avoid obstacles. As illustrated in Figure 8(d), the manipulator encountered obstacles at about 30 per cent of the task.

The null space projection method is based on formula (6), the gradient of the distance from obstacles is projected into the null space of the manipulator's Jacobian matrix, so the manipulator would be "pushed" away from the obstacles on self-motion manifolds. However, this kind of local optimization method cannot handle local minimum, as shown in Figure 8(e); the manipulator was "pushed" away from the first obstacle, but it encounters local minimum at about 40 per cent of the given path.

Figure 8 (a) Experiment B: collision avoidance; (b) the path of end-effector in experiment B; (c) the start configuration; (d) the final configuration of Jacobian pseudo-inverse method; (e) the final configuration of null space projection method; (f) the final configuration of the proposed method

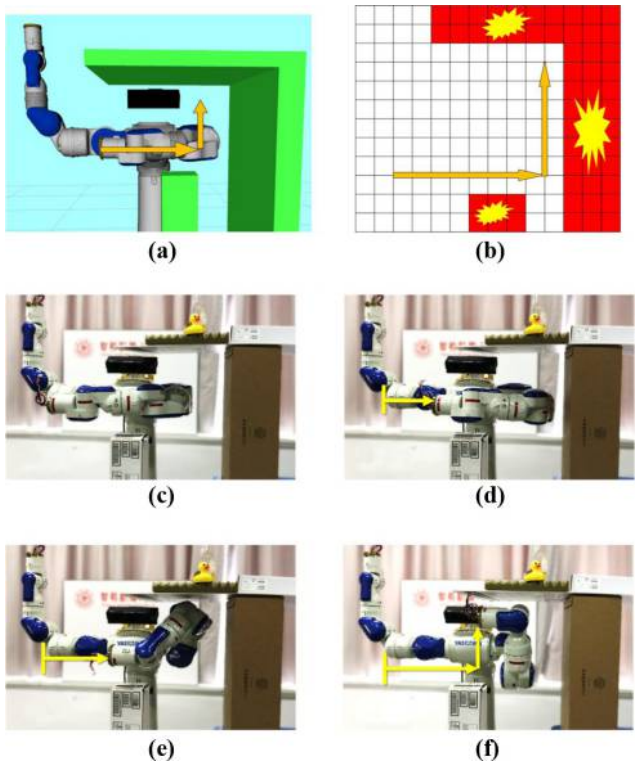


Figure 9 Experiment C: car door polishing task in simulation, the postures of the car door in this experiment is randomly specified in each episode and the polishing path is fixed on the car door

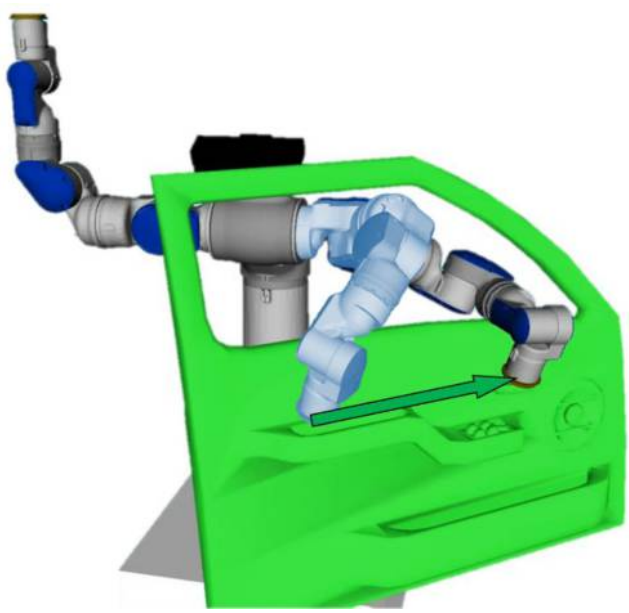


Table I The results of experiment C

The experiment results of the car door polishing task in simulation	Discretization resolution		Planning time (ms)	Tool path error (mm)	Success rate (%)
Jacobian	–		0.15	–	0
NS projection	–		5.34	0.013	15
Our method #1	$\Delta s = 20mm$	$\Delta\varphi = 0.2rad$	136.35	1.32	85
Our method #2	$\Delta s = 5mm$	$\Delta\varphi = 0.2rad$	837.86	0.353	93
Our method #3	$\Delta s = 20mm$	$\Delta\varphi = 0.05rad$	936.32	0.532	91
Our method #4	$\Delta s = 5mm$	$\Delta\varphi = 0.05rad$	6356.76	0.021	98

As for our method, the task constrained manifolds are explicitly converted into a digraph, so it can find a valid path according to the global information of the environment. As shown in Figure 8(f), the left arm of the Motoman SDA can avoid all obstacles while moving along the given end-effector path.

5.3 Performance evaluation

The previous two experiments only demonstrate the difference among our method and the others, so we set up the third experiment to quantificationally evaluate the performance of our method. As shown in Figure 9, the left hand of Motoman SDA should following a specified path along the car door model, while the postures of the car door were randomly specified in each episode. Six methods are implemented: simple Jacobian pseudo-inverse method, null space projection method and our method with four kinds of discretization resolution.

For each experiment, the planning time, success rate and tool path error are record, and all values were averaged over 100 planner executions. The machine we used had an Intel i7-4710HQ CPU (2.50 GHz) and 16 GB RAM. All experiments are single threaded.

As we can see from Table I, the Jacobian-based method is very fast and accurate, because we can use direct velocity control in simulation environment, but the success rate is not satisfactory, as both these two methods often fail in finishing this task. Besides, our methods are good at finding a valid joint trajectory to fulfill this task but require more planning time. What's more, the discretization resolution is critical to the performance of the proposed method, a higher discretization resolution will lead to a larger planning time and higher accuracy and vice-versa. For a real application, we can choose a proper discretization resolution according to the process requirement.

6. Conclusions and future work

We proposed an explicit expression of the task submanifolds for a 7-DOF redundant manipulator. The submanifolds is divided into Cartesian distance dimension s and self-motion dimension φ, all configurations on the task manifolds can be accessed by two parameters. Therefore, the task manifolds can be converted into a directed graph based on the parameterization method, and any sub-constraints, such as collision and singularity avoidance, can be applied to modify the directed graph. Especially, an error estimation method is introduced in the graph connection section, to reduce the step error. Finally, the global graph search algorithms can be used to find a valid joint path in the directed graph.

Many industrial applications are defined as tool motions, so this method can be used in this field to reduce time spent on human teaching. Therefore, the next work would be applying this algorithm to robot polishing and other industrial tasks.

Besides, master/slave-based dual arm cooperation is also a candidate application of this algorithm. The motion of slave arm is restricted on the low dimensional task submanifolds, so future work can be using this method in dual arm cooperative motion planning tasks.

What's more, this algorithm can be improved by applying parallel collision detection method (Pan and Manocha, 2012) or lazy collision detection (Hauser, 2015), adopting new constraints and implementing more effective graph search algorithms.

Additionally, a formal proof of resolution completeness and resolution optimality property should be included in our future work.

References

Berenson, D., Srinivasa, S. and Kuffner, J. (2011), "Task space regions: a framework for pose-constrained manipulation planning", *The International Journal of Robotics Research*, Vol. 30 No. 12, pp. 1435-1460.

Berenson, D., Srinivasa, S.S., Ferguson, D. and Kuffner, J.J. (2009), "Manipulation planning on constraint manifolds", *IEEE International Conference on Robotics and Automation, 2009, ICRA'09, IEEE*, pp. 625-632.

Dijkstra, E.W. (1959), "A note on two problems in Connexion with graphs", *Numerische Mathematik*, Vol. 1 No. 1, pp. 269-271.

Dubey, R.V., Euler, J.A. and Babcock, S.M. (1991), "Real-time implementation of an optimization scheme for seven-degree-of-freedom redundant manipulators", *IEEE Transactions on Robotics and Automation*, Vol. 7 No. 5, pp. 579-588.

Flacco, F., De Luca, A. and Khatib, O. (2012), "Motion control of redundant robots under joint constraints: saturation in the null space", *2012 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp. 285-292.

Hart, P.E., Nilsson, N.J. and Raphael, B. (1968), "A formal basis for the heuristic determination of minimum cost paths", *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4 No. 2, pp. 100-107.

Hauser, K. (2015), "Lazy collision checking in asymptotically-optimal motion planning", *2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp. 2951-2957.

Kamiya, Y., Asahi, T., Ando, S. and Khatib, O. (2013), "Task oriented control with constraints for industrial robot", *2013 44th International Symposium on Robotics (ISR)*, IEEE, pp. 1-6.

Kavraki, L.E., Svestka, P., Latombe, J.-C. and Overmars, M.H. (1996), "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", *IEEE Transactions on robotics and Automation*, Vol. 12 No. 4, pp. 566-580.

Khatib, O. (1987), "A unified approach for motion and force control of robot manipulators: the operational space formulation", *IEEE Journal on Robotics and Automation*, Vol. 3 No. 1, pp. 43-53.

Kim, B., Um, T.T., Suh, C. and Park, F.C. (2016), "Tangent bundle RRT: a randomized algorithm for constrained motion planning", *Robotica*, Vol. 34 No. 1, pp. 202-225.

Kobayashi, S. and Nomizu, K. (1996), *Foundations of Differential Geometry*, Interscience, New York, NY, Google Scholar, Vol. 2, p. 261.

Kucuk, S. (2016), "Maximal dexterous trajectory generation and cubic spline optimization for fully planar parallel manipulators", *Computers & Electrical Engineering*, Vol. 56, pp. 634-647.

LaValle, S.M. (1998), "Rapidly-exploring random trees: a new tool for path planning".

Maciejewski, A.A. and Klein, C.A. (1985), "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments", *The international Journal of Robotics Research*, Vol. 4 No. 3, pp. 109-117.

Nakamura, Y., Hanafusa, H. and Yoshikawa, T. (1987), "Task-priority based redundancy control of robot manipulators", *The International Journal of Robotics Research*, Vol. 6 No. 2, pp. 3-15.

Pan, J. and Manocha, D. (2012), "GPU-based parallel collision detection for fast motion planning", *The International Journal of Robotics Research*, Vol. 31 No. 2, pp. 187-200.

Pan, J., Chitta, S. and Manocha, D. (2012), "FCL: a general purpose library for collision and proximity queries", *2012 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp. 3859-3866.

Sadeghian, H., Villani, L., Keshmiri, M. and Siciliano, B. (2014), "Task-space control of robot manipulators with null-space compliance", *IEEE Transactions on Robotics*, Vol. 30 No. 2, pp. 493-506.

Shen, H., Wu, H., Chen, B., Jiang, Y. and Yan, C. (2015), "Obstacle avoidance algorithm for 7-DOF redundant anthropomorphic arm", *Journal of Control Science and Engineering*, Vol. 2015, p. 7.

Shimizu, M., Kakuya, H., Yoon, W.-K., Kitagaki, K. and Kosuge, K. (2008), "Analytical inverse kinematic computation for 7-DOF redundant manipulators with joint limits and its application to redundancy resolution", *IEEE Transactions on Robotics*, Vol. 24 No. 5, pp. 1131-1142.

Suh, K. and Hollerbach, J. (1987), "Local versus global torque optimization of redundant manipulators", *1987 IEEE International Conference on Robotics and Automation, Proceedings*, IEEE, pp. 619-624.

Yakey, J.H., LaValle, S.M. and Kavraki, L.E. (2001), "Randomized path planning for linkages with closed kinematic chains", *IEEE Transactions on Robotics and Automation*, Vol. 17 No. 6, pp. 951-958.

Yao, Z. and Gupta, K. (2005), "Path planning with general end-effector constraints: using task space to guide configuration space search", *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005. (IROS 2005)*, IEEE, pp. 1875-1880.

Yoshikawa, T. (1984), "Analysis and control of robot manipulators with redundancy", *Robotics Research: The First International Symposium*, Mit Press Cambridge, MA, pp. 735-747.

Yoshikawa, T. (1985), "Manipulability and redundancy control of robotic mechanisms", *1985 IEEE International Conference on Robotics and Automation. Proceedings*, IEEE, pp. 1004-1009.

Corresponding author

Qiang Qiu can be contacted at: qiu6401@sjtu.edu.cn