Optimal Joint Trajectory Planning for Manipulator Robot Performing Constrained Motion Tasks

Yueshi Shen
Department of Information Engineering
Australian National University
Canberra ACT 0200, Australia
yueshi.shen@anu.edu.au

Knut Hüper
National ICT Australia Limited *
Locked Bag 8001
Canberra ACT 2601, Australia
knut.hueper@nicta.com.au

Abstract

This paper presents a novel approach to plan the optimal joint trajectory for a manipulator robot performing constrained motion tasks. In general, a two-step scheme will be deployed to find the optimal robot joint curve. Firstly, instead of solving a nonlinear, implicit Euler-Lagrange equation, we discretize the corresponding cost function and use Newton's iterations to numerically calculate the joint trajectory's intermediate discrete points. Secondly, we interpolate these points to get the final joint trajectory in a way such that the motion constraint will always be sustained throughout the movement. An example of motion planning for a 4-degree-of-freedom robot WAM will be given at the end of this paper.

1 Introduction

Generally speaking, the task for a motion planner is to specify a motion to be executed by the actuator. A proper motion plan can have advantages with respect to different aspects, for example, obstacle avoidance, work efficiency optimization, better tracking performance etc. For multi-link robotic systems, the automatic task execution can be divided into three smaller subproblems [Singh and Leu, 1991]:

P1 For a given robot and task, plan a path for the end-effector between two specified positions. Such a path optimizes a performance index, in the meantime satisfies either equality (for instance, robot's end-tip is required to move on a surface) or inequality (for instance, obstacle avoidance, joint angle limit) constraints.

*National ICT Australia is funded by the Australian Department of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Centre of Excellence Programs.

P2 For a given end-effector path expressed in the task (operational) space (usually coincides with the Cartesian space), find the joint trajectory according to our knowledge about the robot kinematics (and dynamics when necessary). Similarly, some performance index can be optimized in case of a redundant robot, namely, the robot has more degrees of freedom (DOFs) than necessary to perform the given task.

P3 Design a feedback controller which can track the given reference joint trajectory accurately.

P1 and P2 are often resolved separately in robot path planning researches. Especially for P1, to design a collision-free path for a single rigid object (free-flying object) travelling in a crowded environment is a widely researched topic. Latombe's work collects most of such algorithms [Latombe, 1991].

Solving subproblem P2 usually involves the usage of the mathematical relationship between Cartesian and joint coordinates, i.e., the Jacobian. For a redundant robot, the extra DOFs can be utilized to achieve some optimizations besides fulfilling the basic task of end-effector path tracking. Yoshikawa proposed a formulation of joint velocity using the pseudo-inverse of the Jacobian for optimizing a local cost function [Yoshikawa, 1990]. For finding a global optimal trajectory, variational calculus is commonly used. The predefined end-effector path is considered as a constraint to the optimization procedure, i.e., $\mathbf{p}_d(t) - \kappa(\mathbf{q}(t)) = 0$, where $\mathbf{p}_d(t)$ is the desired end-effector path, $\mathbf{q}(t)$ is the joint trajectory to be calculated, and κ denotes the forward kinematic position map. Then by eliminating the Lagrange multiplier, a differential equation in the joint variables can be derived. The joint trajectory are computed by numerically solving the differential equation with boundary conditions [Martin *et al.*, 1989] [Agrawal and Xu, 1994].

More recent work prefers combining the path and trajectory planning (P1 and P2) and takes the dynamics model into consideration, so the robot path planning can

be interpreted as an optimal control problem. This approach automatically eliminates the necessity of calculating the feasible configuration space for the manipulator which is not a trivial problem for a high-DOF robot arm subject to joint limits. Also, by using such path planning strategy, it's easy to formulate some dynamic optimization criterions, e.g., minimum execution time, or incorporate some dynamic constraints, e.g., drive torque limit. In the optimization process, the time history of joint drive torque (or joint position, joint velocity) is usually approximately represented by a limited number of parameters (for example, discretized points, B-spline etc.), so that nonlinear programming or sequential quadratic programming technique can further be used to find the optimal solution [Singh and Leu, 1991], [Wang and Hamam, 1992], [Wang *et al.*, 2001], [Lo Bianco and Piazzi, 2002].

However, the above path planner usually requires calculating the derivative of robot inverse dynamics, which will be quite complicated and time-consuming for a high-DOF robot. Moreover, besides the multi-rigid-body model, other factors which are either difficult to model (e.g., motor torque ripple) or not a smooth function (e.g., Coulomb friction) also account for a considerable proportion of the robot dynamics. Thus it's not so meaningful to compute an "optimal" time history of control input and expect such open-loop control law will end up with optimal performance on the real robotic system.

This paper studies the optimization of the robot path in joint space (solving P1 and P2 at the same time) with regard to some geometric performance index for both robot's joint and end-effector. Furthermore, the optimal joint trajectory is subject to some motion constraint with respect to the end-effector. Because of the synthetical optimality and the motion constraint, the cost function incorporates the robot kinematics and the geometry of environment, for which fairly accurate models are possible to obtain. Dynamic constraints such as limit of joint velocity or drive torque can possibly be treated by using proper time scaling techniques (e.g., dynamic trajectory scaling [Sciavicco and Siciliano, 1996]) afterwards. The rest of this paper is organized as follows: Section 2 gives the mathematical formulation of our optimal robot path planning problem; Section 3 presents a two-step calculation scheme which computes the optimal joint trajectory; Section 4 shows an example of applying our algorithm for planning the motion for a 4-DOF robot WAM with its end-effector is constrained to move on a sphere.

2 Problem Description

Firstly, we are considering the situation that the robot is subject to some holonomic motion constraint

$$C(\mathbf{q}(t)) = 0 \tag{1}$$

where $\mathbf{q} : \mathbb{R} \to \mathbb{R}^p$ is the joint position curve.

For example, if the manipulator's end-effector is required to move on a 2D plane in $\mathbb{R}^3$ (Fig. 1), the corresponding motion constraint can be expressed as:

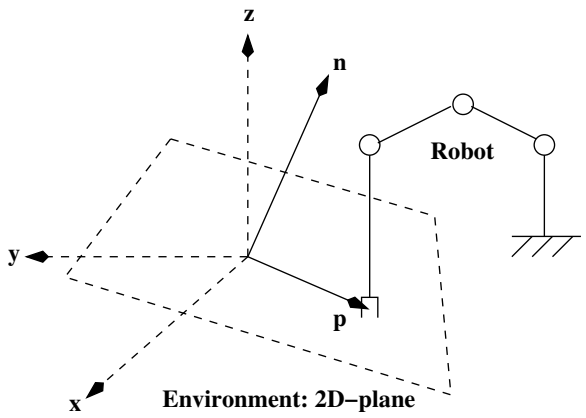

Figure 1: Robot's end-effector is required to move on a 2D-plane

$$\begin{aligned} C(\mathbf{q}(t)) &= \kappa(\mathbf{q}(t))^T \mathbf{n} \\ &= \mathbf{p}(t)^T \mathbf{n} \end{aligned} \tag{2}$$

where $\mathbf{p} \in \mathbb{R}^3$ is the position of the end-effector, $\mathbf{n} \in \mathbb{R}^3$ is the plane normal, κ is robot forward kinematic position map.

Now define the optimization problem as:

P4 Over the set of sufficiently smooth curves subject to motion constraint and boundary conditions:

$$\begin{gathered} \forall t \in [t_0, t_n], C(\mathbf{q}(t)) = 0 \\ \mathbf{q}(t_0) = \mathbf{q}_0,\ \mathbf{q}(t_n) = \mathbf{q}_n \end{gathered} \tag{3}$$

Find a joint trajectory $\mathbf{q} : [t_0, t_n] \to \mathbb{R}^p$, which minimizes the cost function F

$$F(\mathbf{q}(t), \dot{\mathbf{q}}(t)) = \int_{t_0}^{t_n} f\big(\mathbf{q}(t), \dot{\mathbf{q}}(t)\big)\, \mathrm{d}t \tag{4}$$

where $f : [t_0, t_n] \to \mathbb{R}$ is the Lagrangian.

By introducing the Lagrange multiplier function $\mu : [t_0, t_n] \to \mathbb{R}$, the original optimization problem **P4** is equivalent to:

P5 Find smooth curves $\mathbf{q}(t)$ and $\mu(t)$, so that the following function $\tilde{F}$ is stationary (minimal).

$$\tilde{F}(\mathbf{q}, \dot{\mathbf{q}}, \mu) = \int_{t_0}^{t_n} \Big(f(\mathbf{q}, \dot{\mathbf{q}}) + \mu C(\mathbf{q}) \Big) \mathrm{d}t \tag{5}$$

Using variational calculus, we know that the optimal joint curve for the above path planning problem satisfies an Euler-Lagrange equation

$$\begin{aligned} \frac{\partial f}{\partial \mathbf{q}} - \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial f}{\partial \dot{\mathbf{q}}} + \mu \frac{\mathrm{d}C}{\mathrm{d}\mathbf{q}} &= 0 \\ C(\mathbf{q}(t)) &= 0 \end{aligned} \tag{6}$$

Ideally, we can try to solve for $\mathbf{q}(t)$ from Eq. 6 by eliminating the Lagrange multiplier $\mu(t)$. However, the cost function may incorporate the nonlinear robot kinematics model as the example in Section 4, so that the differential equation for $\mathbf{q}(t)$ will become implicit. In the next section, we will discuss an alternative approach which first computes the intermediate points and then interpolate them to form the final curve of $\mathbf{q}(t)$.

3 Solution

To accomplish a path optimization numerically, the usual approach is to approximately represent the resulting curve by a finite number of parameters. In this paper, we use the curve's discrete positions only, from which reasonable approximations of velocity and acceleration can be obtained as long as the original curve doesn't contain too much high-frequency oscillation. Optimization is considered in calculating the intermediate discrete points, which can be shown to converge to the result of Euler-Lagrange equation. Later, these points will be interpolate by a smooth curve as a suboptimal solution for the problem **P4**.

3.1 Discretization Scheme

Consider discretizing the time interval $[t_0, t_n]$ by a regular partition

$$t_k = kh \ , \ k \in \{0, 1, \cdots, n-1, n\} \tag{7}$$

where $h = (t_n - t_0)/n$ is the step size.
Let

$$\mathbf{q}_k := \mathbf{q}(t_k) \ , \ k \in \{0, 1, \cdots, n-1, n\} \tag{8}$$

Given the boundary values $\mathbf{q}_0$ and $\mathbf{q}_n$, we wish to compute the set of the intermediate points $\{\mathbf{q}_1 \cdots \mathbf{q}_{n-1}\}$.
Similarly, define

$$\mu_k := \mu(t_k) \ , \ k \in \{0, 1, \cdots, n-1, n\} \tag{9}$$

3.2 Integration and Approximation Scheme

Consider the following integration scheme for $\tilde{F}$ defined by Eq. 5:

$$\begin{aligned} \tilde{F}(\mathbf{q}, \dot{\mathbf{q}}, \mu) \quad \simeq \quad & \sum_{k=0}^{n-1} f(\mathbf{q}(\tau_k), \dot{\mathbf{q}}(\tau_k))h + \\ & \mu_0 C(\mathbf{q}_0)\frac{h}{2} + \sum_{k=1}^{n-1} \mu_k C(\mathbf{q}_k)h + \mu_n C(\mathbf{q}_n)\frac{h}{2} \end{aligned} \tag{10}$$

where

$$\tau_k := kh + \frac{h}{2} \ , \ k \in \{0, 1, \cdots, n-1\} \tag{11}$$

Furthermore, $\mathbf{q}(\tau_k)$ and $\dot{\mathbf{q}}(\tau_k)$ can be approximated as:

$$\mathbf{q}(\tau_k) \simeq \frac{\mathbf{q}_k + \mathbf{q}_{k+1}}{2} \tag{12}$$

$$\dot{\mathbf{q}}(\tau_k) \simeq \frac{\mathbf{q}_{k+1} - \mathbf{q}_k}{h} \tag{13}$$

By combining Eqs. 10, 12, 13, we approximate $\tilde{F}$ as a function of the discretized joint position $\mathbf{q}_k$ and Lagrange multiplier μ_k. The variational problem **P5** is now converted into a finite-dimensional optimization problem **P6**:

P6 Find a vector $\mathbf{Q} \in \mathbb{R}^s, s = (n-1)(p+1)$

$$\mathbf{Q} = \Big[q_{1,1} \ , \ \cdots \ , \ q_{n-1,p} \ , \ \mu_1 \ , \ \cdots \ , \ \mu_{n-1} \Big]^T \tag{14}$$

where $q_{k,j}$ is the jth element of $\mathbf{q}_k$.
which minimizes the function $\tilde{F}'$

$$\begin{aligned} \tilde{F}'(\mathbf{Q}) = \quad & \sum_{k=0}^{n-1} f(\frac{\mathbf{q}_k + \mathbf{q}_{k+1}}{2}, \frac{\mathbf{q}_{k+1} - \mathbf{q}_k}{h}) + \\ & \sum_{k=1}^{n-1} \mu_k C(\mathbf{q}_k) \end{aligned} \tag{15}$$

To solve **P6**, we will apply Newton's method to iteratively calculate $\mathbf{Q}$.

A1 (Newton iteration)

1. Pick a reasonable guess of $\mathbf{Q}$.
2. Update $\mathbf{Q}$ by the following law:

$$\mathbf{Q}_{i+1} = \mathbf{Q}_i - \mathbf{H}^{-1}(\mathbf{Q}_i)\nabla\tilde{F}'(\mathbf{Q}_i) \tag{16}$$

where $\mathbf{Q}_i$ is the ith iterate of $\mathbf{Q}$, $\nabla\tilde{F}'$ is the gradient of $\tilde{F}'$ with respect to $\mathbf{Q}$, and $\mathbf{H}$ is the matrix of its 2nd derivative (Hessian).
Apply step 2 until the norm of $\nabla\tilde{F}'$ gets small enough.

The integration/approximation scheme used above preserves the characteristics of the original optimization system. Levin *et al.* proved that the solution of the discretized system will converge to the solution of Euler-Lagrange equation as h goes to 0 [Levin *et al.*, 2002].

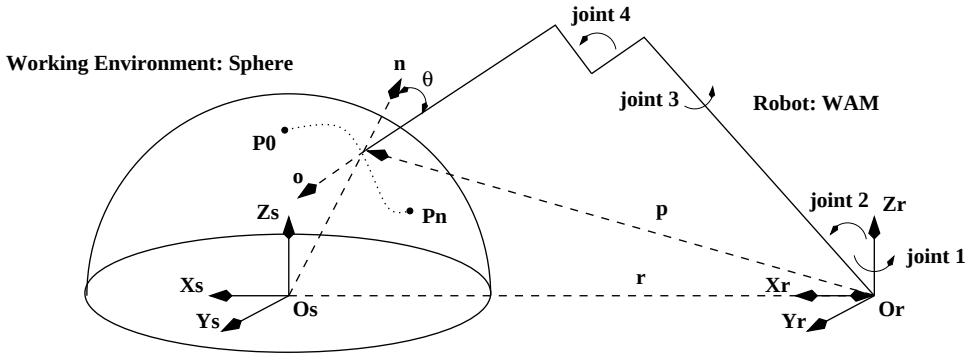

Figure 2: WAM moves its end-effector on a sphere

3.3 Interpolation

Having computed the intermediate points, the next step is to find out the joint trajectory $\mathbf{q}(t)$ such that

$$\mathbf{q}(t_k) = \mathbf{q}_k \quad (17)$$

$$\forall t \in [t_0, t_n],\ C(\mathbf{q}(t)) = 0 \quad (18)$$

To ensure the resulting joint curve satisfies both two conditions (Eqs. 17, 18), we design a 3-step calculation scheme (algorithm **A2**) as follows:

A2

1. Interpolate the boundary and computed intermediate points $\mathbf{q}_0, \mathbf{q}_1 \cdots \mathbf{q}_{n-1}, \mathbf{q}_n$ by a cubic spline. We call the interpolating curve $\mathbf{q}_{org}(t)$.
2. Interpolate $\mathbf{p}_k = \kappa(\mathbf{q}_k)$ by a curve $\mathbf{p}(t)$ on the working surface. A pull back/push forward technique with rolling and wrapping for smooth interpolating curves on a manifold was proposed recently [Hüper and Silva Leite, 2002]. This can be used to compute $\mathbf{p}(t)$.
3. Adjust the joint trajectory $\mathbf{q}_{org}(t)$ to fit the end-effector path $\mathbf{p}(t)$ by repeatedly applying the following law (initially, set $\mathbf{q}_{old}(t) = \mathbf{q}_{org}(t)$)

$$\mathbf{q}_{new}(t) = \mathbf{q}_{old}(t) + \mathbf{J}_v^{\dagger}(\mathbf{q}_{old}(t))\big(\mathbf{p}(t) - \kappa(\mathbf{q}_{old}(t))\big) \quad (19)$$

where $\mathbf{J}_v$ is the $3 \times p$ matrix mapping joint velocity to end-effector's linear velocity, and $\mathbf{J}_v^{\dagger}$ is its pseudo inverse.

Using the pseudo-inverse of the Jacobian will fine-tune the robot pose to make the end-effector position get closer to the desired path $\mathbf{p}(t)$ with minimum change of joint position ($\triangle\mathbf{q}(t) = \mathbf{q}_{new}(t) - \mathbf{q}_{old}(t)$). Here we assume we've already have enough intermediate points, so that $\mathbf{p}(t)$ and $\kappa(\mathbf{q}_{org}(t))$ will be sufficiently close to each other.

4 Example

WAM is a 4-joint robot manipulator with human-like kinematics (see Fig. 3). Now suppose the robot is required to move its end-effector on some working surface, for instance, a sphere (see Fig. 2). Viewed in Cartesian space, WAM's 4-DOF corresponds to its end-effector moving to arbitrary 3D position and rotating about vector $\mathbf{p}$ in Fig. 2 (1-DOF in orientation).

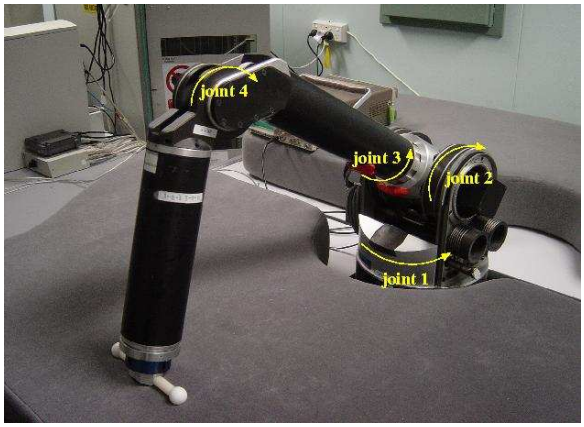

Figure 3: 4-DOF robot WAM

Given the initial and final joint positions (which can be arbitrary as long as the motion constraint is satisfied) and assuming the whole sphere surface is within WAM's workspace, now the task is to find an optimal joint trajectory yielding a comprehensive optimality with respect to the following three aspects.

1. The last link should be kept perpendicular to the sphere as much as possible (i.e., $\theta \rightarrow \min$). Perpendicularity will have advantages in many situations, e.g., spray coating, force sensing etc.
2. The end-effector traverses minimum distance.
3. The joints traverse minimum distance.

The overall cost function of the variational problem can be given as:

$$\mathbf{F}(\mathbf{q},\dot{\mathbf{q}}) = \int_0^1 \Big(\frac{\mathbf{o}^T(\mathbf{r}+\mathbf{p})}{\|\mathbf{o}^T(\mathbf{r}+\mathbf{p})\|} + \alpha(\dot{\mathbf{p}}^T\dot{\mathbf{p}}) + \beta(\dot{\mathbf{q}}^T\dot{\mathbf{q}}) \Big) \mathrm{d}t \tag{20}$$
$$s.t. \quad \|\mathbf{r}+\mathbf{p}(t)\| = R,\ \mathbf{q}(0) = \mathbf{q}_0,\ \mathbf{q}(1) = \mathbf{q}_n$$

where $\mathbf{o}(\mathbf{q}(t)) \in \mathbb{R}^3$ represents the orientation of WAM's last link, $\mathbf{p}(\mathbf{q}(t))$ and $\dot{\mathbf{p}}(\mathbf{q}(t),\dot{\mathbf{q}}(t)) \in \mathbb{R}^3$ are respectively its end-effector's position and linear velocity, they are both calculated through WAM's forward kinematics model. α and β are weights of the second and third term. $\mathbf{r}$ is the vector from the center of sphere to the base of WAM, R is the radius of the sphere, and $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}}$. Vectors $\mathbf{o}$, $\mathbf{p}$ and $\mathbf{r}$ are as Fig. 2 illustrates.

Figs. $4-7$ show the output of step 1. Here we compare the optimal result for three choices of α and β in the cost function (Eq. 20):

1. $\alpha = 0.5, \beta = 0$
   consider orientation, end-effector path, but not joint trajectory.
2. $\alpha = 0, \beta = 0.003$
   consider orientation, joint trajectory, but not end-effector path.
3. $\alpha = 0.5, \beta = 0.003$
   consider orientation, end-effector path, and joint trajectory.

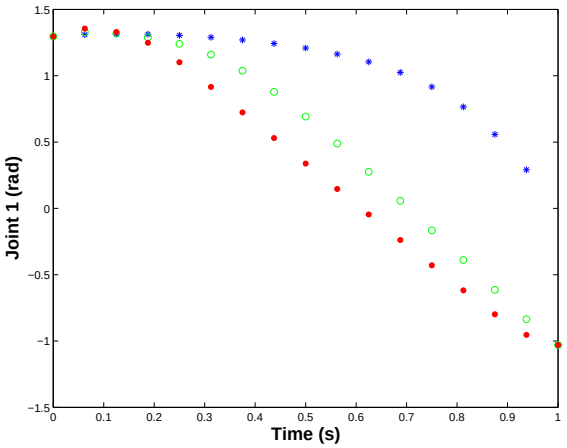

Figure 4: Calculated points of the 1st joint's trajectory Blue: $\alpha = 0.5, \beta = 0$; Red: $\alpha = 0, \beta = 0.003$; Green: $\alpha = 0.5, \beta = 0.003$

As we can see in Figs. 8, 9, 10, all end-effector points (which are computed from the joint points by forward kinematics) are located on the sphere, and they tend

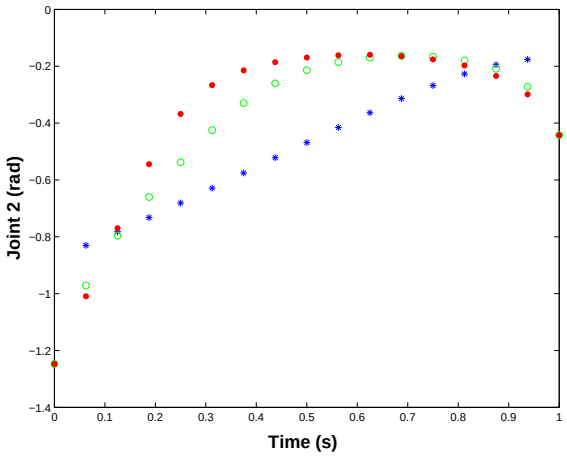

Figure 5: Calculated points of the 2nd joint's trajectory Blue: $\alpha = 0.5, \beta = 0$; Red: $\alpha = 0, \beta = 0.003$; Green: $\alpha = 0.5, \beta = 0.003$

to move close to the black line which indicates the only place that WAM's last link is possible to be perpendicular to the surface. The 'incomplete' optimization criterion in case 1 and 2 give rise to some 'abrupt hop' for either joint trajectory (Figs. 4blue, 5blue, 6blue) or end-effector path (Fig. 9).

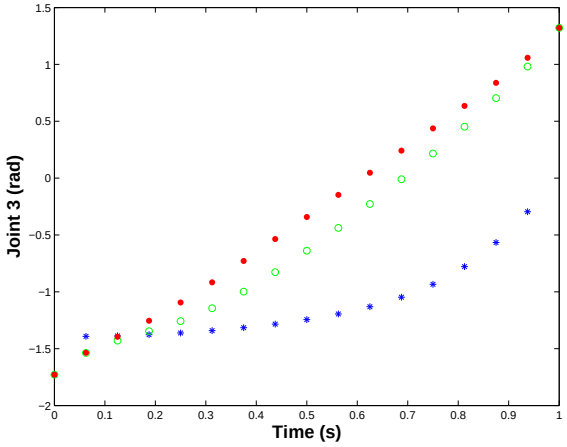

Figure 6: Calculated points of the 3rd joint's trajectory Blue: $\alpha = 0.5, \beta = 0$; Red: $\alpha = 0, \beta = 0.003$; Green: $\alpha = 0.5, \beta = 0.003$

Figs. 11, 12, 13 show the interpolating curve for the result in Figs. 4green, 5green, 6green, 10. By using the algorithm **A2** described in Section 3.3, we can see the final joint trajectory satisfies the requirements Eqs. 17, 18, also both the end-effector path and joint trajectory are smooth.

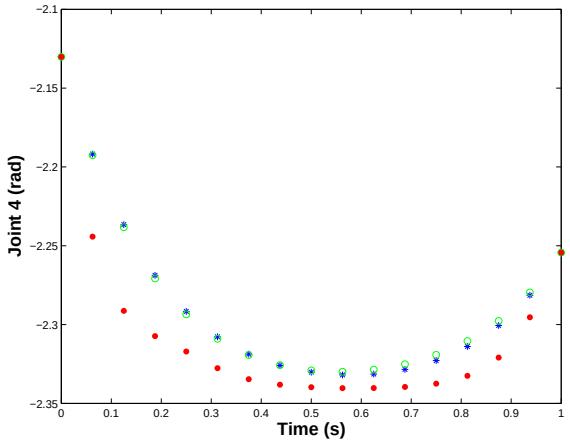

Figure 7: Calculated points of the 4th joint's trajectory Blue: $\alpha = 0.5, \beta = 0$; Red: $\alpha = 0, \beta = 0.003$; Green: $\alpha = 0.5, \beta = 0.003$

5 Conclusion

This paper presents a method to calculate the optimal joint trajectories for manipulator robot performing constrained motion task. The path planning are completed in two stages: 1) convert the variational calculus problem into a finite dimensional optimization problem, use Newton's method to compute the joint trajectory's intermediate points; 2) interpolate these points to form the joint curve and adjust it to ensure the motion constraint will be satisfied throughout the entire trajectory.

A Java 3D visualization demo is also submitted with this paper and a motion control experiment on WAM executing the above computed joint trajectory is under consideration.

References

[Agrawal and Xu, 1994] O.P. Agrawal, Y.S. Xu. On the Global Optimum Path Planning for Redundant Space Manipulators. *IEEE Transactions on Systems Man and Cybernetics*, 24(9): 1306-1316 Sep 1994.

[Doggett *et al.*, 2000] W.R. Doggett, W.C. Messner, J.N. Juang. Global Minimization of the Robot Base Reaction Force During 3-D Maneuvers. *IEEE Transactions on Robotics and Automation*, 16(6): 700-711 Dec 2000.

[Hüper and Silva Leite, 2002] K. Hüper, F. Silva Leite. Smooth Interpolating Curves with Applications to Path Planning. In *Proceedings of the 10th Mediterranean Conference on Control and Automation*, Lisbon, Portugal, July 2002.

[Latombe, 1991] J.-C. Latombe. *Robot Motion Planning.* Klumer, Boston, 1991.

[Levin *et al.*, 2002] Y. Levin, M. Nediak, A. Ben-Israel. A Direct Newton Method for Calculus of Variations. *Journal of Computational and Applied Mathematics*, 139(2): 197-213 Feb 15 2002.

[Lo Bianco and Piazzi, 2002] C.G. Lo Bianco, A. Piazzi. Minimum-Time Trajectory Planning of Mechanical Manipulators under Dynamic Constraints. *International Journal of Control*, 75(13): 967-980 Sep 2002.

[Martin *et al.*, 1989] D.P. Martin, J. Baillieul, J.M. Hollerbach. Resolution of Kinematic Redundancy Using Optimization Techniques. *IEEE Transactions on Robotics and Automation*, 5(4): 529-533 Aug 1989.

[Murray *et al.*, 1994] R.M. Murray, Z.X. Li, S.S. Sastry. *A Mathematical Introduction to Robotic Manipulation.* CRC Press, 1994.

[Sciavicco and Siciliano, 1996] L. Sciavicco and B. Siciliano *Modeling and Control of Robot Manipulators.* McGraw-Hill, 1996.

[Singh and Leu, 1991] S.K. Singh, M.C. Leu. Manipulator Motion Planning in the Presence of Obstacles and Dynamic Constraints. *International Journal of Robotics Research*, 10(2): 171-187 Apr 1991.

[Wang *et al.*, 2001] C-Y.E. Wang, W.K. Timoszyk, J.E. Bobrow. Payload Maximization for Open Chained Manipulators: Finding Weightlifting Motions for a Puma 762 Robot. *IEEE Transactions on Robotics and Automation*, 17(2): 218-224 Apr 2001.

[Wang and Hamam, 1992] D. Wang, Y. Hamam. Optimal Trajectory Planning of Manipulators with Collision Detection and Avoidance. *International Journal of Robotics Research*, 11(5): 460-468 Oct 1992.

[Yoshikawa, 1990] T. Yoshikawa. *Foundations of Robotics: Analysis and Control.* MIT Press, Cambridge, Massachusetts, 1990.

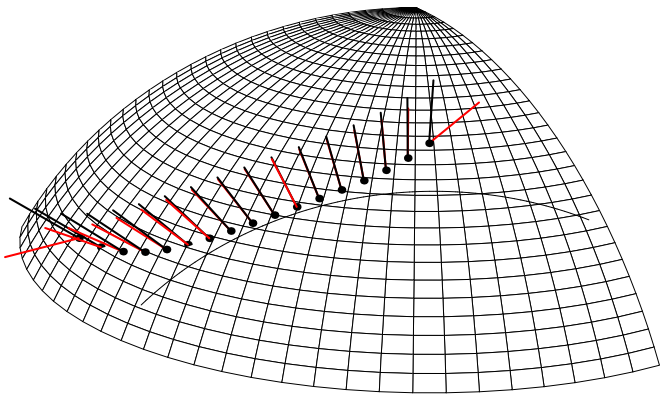

Figure 8: Intermediate points of end-effector path when $\alpha = 0.5, \beta = 0$.
Black: sphere normal; Red: WAM's last link.

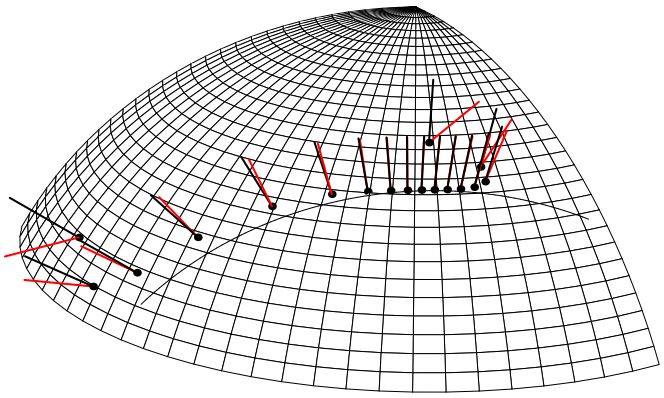

Figure 9: Intermediate points of end-effector path when $\alpha = 0, \beta = 0.003$.
Black: sphere normal; Red: WAM's last link.

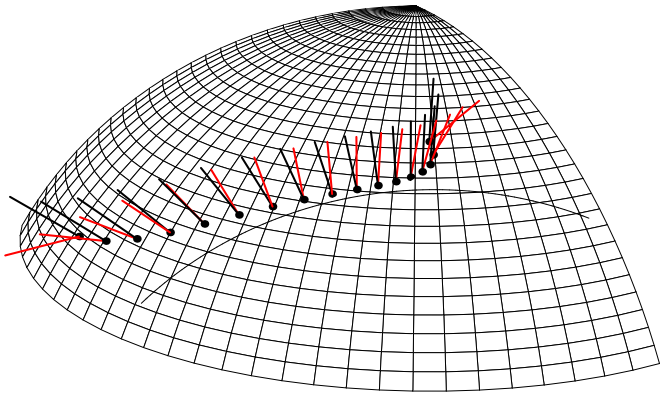

Figure 10: Intermediate points of end-effector path when $\alpha = 0.5, \beta = 0.003$.
Black: sphere normal; Red: WAM's last link.

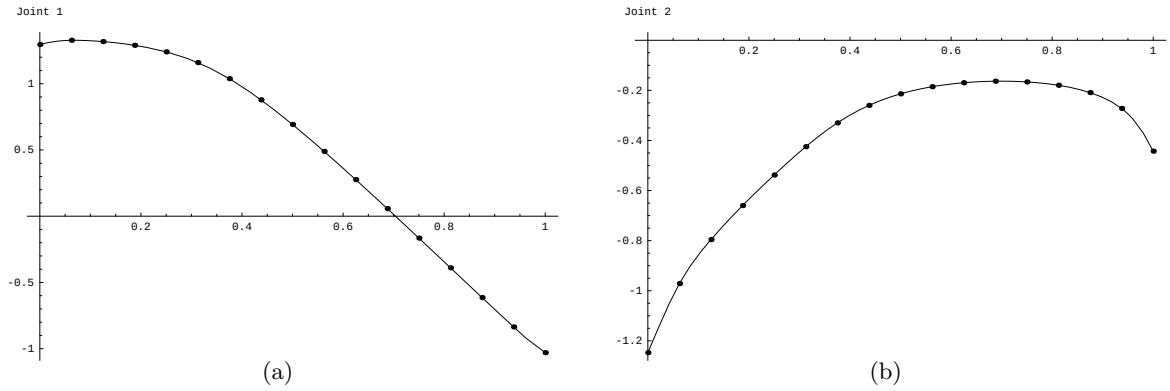

Figure 11: Interpolating joint trajectory, a) joint 1, b) joint 2

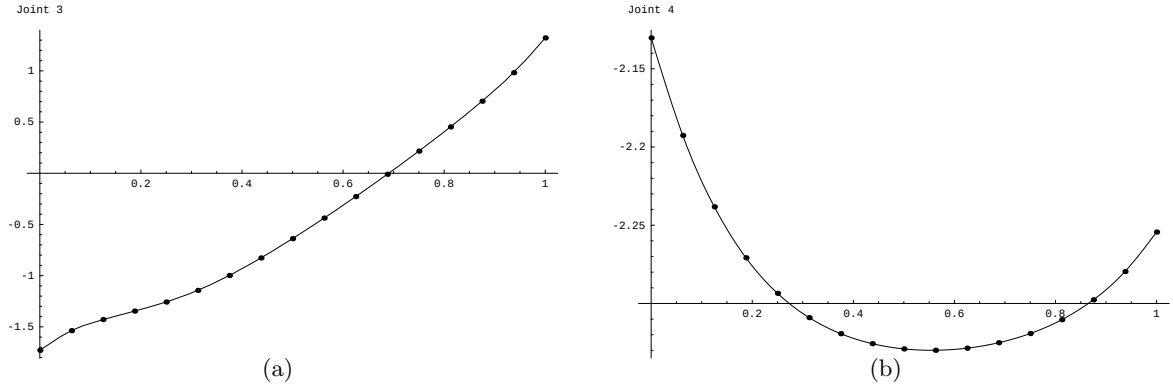

Figure 12: Interpolating joint trajectory, a) joint 3, b) joint 4

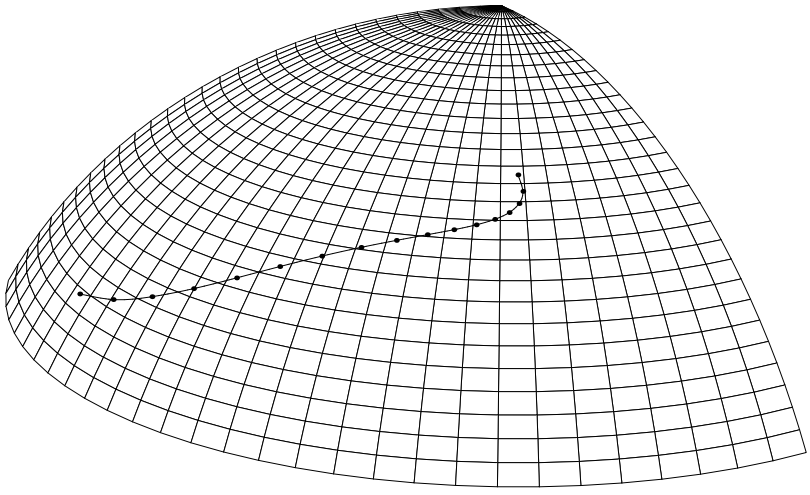

Figure 13: End-effector path calculated from joint trajectory in Figs. 11, 12