Length-Optimized Smooth Obstacle Avoidance for Robotic Manipulators

Soheil S. Parsa[1], Juan A. Carretero[2], Roger Boudreau[3]

[1] *Department of Mechanical Engineering, University of New Brunswick, s.parsa@unb.ca*

[2] *Department of Mechanical Engineering, University of New Brunswick, juan.carretero@unb.ca*

[3] *Department de génie mécanique, Université de Moncton, roger.a.boudreau@umoncton.ca*

Abstract

This paper presents a novel optimised smooth obstacle avoidance algorithm for robotic manipulators. First, a 3-4-5 interpolating polynomial is used to plan a smooth trajectory between initial and final positions in the joint space without considering any obstacles. Then, a simple harmonic function, which is smooth and continuous in displacement, velocity and acceleration, is applied to generate a new smooth path avoiding the obstacles. The obstacle avoidance portions on the path are optimised such that the length of the path traversed by the end-effector is minimised. Simulation results for a 6 DOF serial manipulator demonstrate the efficiency of the proposed method.

Keywords: Robotic Manipulator, Trajectory Planning, Smooth Obstacle Avoidance, Joint space method.

Trajectoire continue à distance minimale pour manipulateurs

Résumé

Cet article présente un nouvel algorithme pour générer des trajectoires continues en évitant des obstacles pour un manipulateur robotique. Un polynôme 3-4-5 est utilisé pour planifier une trajectoire entre la position initiale et la position finale sans considérer les obstacles. Une fonction trigonométrique continue en déplacement, en vitesse et en accélération est ensuite utilisée pour générer une nouvelle trajectoire qui évite les obstacles. L'évitement d'obstacles est optimisé pour minimiser la distance parcourue par l'organe terminal. Des résultats de simulation avec un manipulateur sériel à 6 degrés de liberté démontrent l'efficacité de la méthode proposée.

Mots-clé: Manipulateur robotique, Planification de trajectoire, Evitement d'obstocles, Méthode articulaire.

1 INTRODUCTION

Trajectory planning for robotic manipulators deals with finding a path between the initial and final positions of the end-effector. The designed path needs to be as smooth as possible; *i.e.*, abrupt changes in displacement, velocity and acceleration have to be prevented. Although smooth motions can be produced through relatively simple methods, there is no guarantee that no sudden changes occur in the case of obstacle avoidance. Ambike and Schmiedeler applied curvature theory to track a path for a 2-degree-of-freedom (DOF) planar manipulator [1]. Dash *et al.* proposed a numerical method to generate the reachable workspace of parallel manipulators and for determining the singular points [2]. They also presented a method to find an optimal path over the reachable workspace. Zhou *et al.* investigated the capability of singularity free path generation for five-bar slider-crank parallel manipulators [3]. Bhattacharya *et al.* introduced a method for path planning for parallel manipulators and constrained the manipulator in a single branch to avoid the singularity barriers [4]. Dasgupta and Mruthyunjaya developed an algorithm to generate continuous trajectories in the workspace while avoiding the singularities [5]. Merlet proposed an algorithm which verifies whether a straight line between the initial and final positions of a parallel manipulator is inside the workspace or not [6].

Much research has been undertaken to solve the obstacle avoidance problem for robots (*e.g.*, [7, 8, 9]). Unfortunately, the majority of the developed obstacle avoidance algorithms apply to mobile robots. Also, most of the proposed algorithms implement the free-of-obstacle workspace for the manipulator which dramatically decreases its reachable workspace. Another important issue to consider in the obstacle avoidance problem is the sudden changes in joint acceleration when the pre-designed path has to be changed for collision avoidance. For instance, abrupt changes in acceleration generate infinite jerk spikes which is known to decrease the life of the manipulator's actuators.

In [10], a 3-4-5 polynomial was used to plan a trajectory in joint space between an initial and a final position. Then, a harmonic function which is continuous and smooth in displacement, velocity and acceleration was applied to modify the pre-designed joint trajectory to avoid obstacles in the Cartesian space. According to the obstacle avoidance algorithm by Parsa *et al.*, the acceleration of actuators changes smoothly during the obstacle avoidance section. Here, a similar strategy is used with the addition of an optimization strategy to minimize the length of the end-effector path in the Cartesian space. That is, the method looks at minimizing the end-effector's traversed path during the obstacle avoidance section. To illustrate the method, the proposed strategy is applied to an obstacle avoidance task with a 6 DOF serial manipulator and the results are compared to the strategy without the optimization.

2 TRAJECTORY PLANNING

For the proposed algorithm, a 3-4-5 interpolating polynomial is used to design a trajectory in joint space [9]. Since the applied polynomial can be differentiated twice, the displacement, velocity and acceleration vary continuously. This joint-based polynomial interpolation method only needs the initial and final conditions for every joint position, velocity and acceleration. Considering the number of boundary conditions, the interpolating polynomial method is studied with the aid of a

fifth-order polynomial $S(\tau)$. That is:

$$S(\tau) = a\tau^5 + b\tau^4 + c\tau^3 + d\tau^2 + f \tag{1}$$

where a, b, c, d, e and f are constants and are found through the boundary conditions of the desired motion.

To simplify the calculations for each of the joints, the polynomial in equation (1) is normalized, that is,

$$\tau = \frac{t}{T} \quad \text{and} \quad 0 \leq S \leq 1$$

where t is the actual time, in seconds, T is the time the manipulator takes to fulfil the entire task, and τ is a non-dimensional time parameter (*i.e.*, $0 \leq \tau \leq 1$).

As a result, the polynomial representing the evolution of joint j with respect to the non-dimensional time parameter τ is considered as:

$$\theta_j(t) = \theta_j^I + (\theta_j^F - \theta_j^I)S(\tau) \tag{2}$$

where θ_j^I and θ_j^F are the initial and final positions of joint j ($j = 1 \ldots n$, the number of active joints) obtained by solving the inverse displacement problem on the initial and final end-effector positions, respectively.

In a typical pick and place operation, where conditions of zero end-effector velocity and acceleration at the trajectory's end points need to be satisfied, the velocity and acceleration at each joint are also equal to zero at the start and end of the operation. Consequently, with these boundary conditions set, equation (1) can be written as:

$$S(\tau) = 6\tau^5 - 15\tau^4 + 10\tau^3 \tag{3}$$

In order to design a smooth path between the initial and final points, the total time is first determined. This is often done by considering joint velocities and/or accelerations never exceeding some maximum threshold while applying the equations to the joint with the largest displacement and/or the joint with the smallest velocity or acceleration limits. Once time T is defined for a specific trajectory, sets of joint values which represent the joint displacements are generated using equations (2) and (3).

Although, the 3-4-5 polynomial trajectory planning methods described above yields a smooth path between the initial position to the final one, there is no guarantee that collisions are avoided throughout the trajectory. This turns this method into an unreliable approach when the manipulator's workspace has obstacles.

3 Smooth Obstacle Avoidance

Since the trajectory planning method described above may only meet two specific Cartesian positions without collision (namely the initial and final position), there is a need to modify the predesigned trajectory when obstacles are present. This could be done by re-defining the trajectory in the Cartesian space (using via points for instance [11]). However, depending on the number of via

points used, the resulting polynomial required could be of high order thus producing undesirable oscillations.

Here, as it was done in [10], continuity in the joint trajectories is guaranteed by adding the value of a smooth and continuous function to the pre-designed 3-4-5 joint trajectory. The smooth function is a set of harmonic equations that describe cam motion and consist of four quarter-cycloidal equations as follows:

$$S'(\tau') = \left[\tau' - \frac{1}{\pi}\sin\pi\tau'\right] \quad , \quad 0 \leq \tau' \leq 1 \tag{4}$$

$$S'(\tau') = \left[\tau' + \frac{1}{\pi}\sin\pi\tau'\right] \quad , \quad 1 \leq \tau' \leq 2 \tag{5}$$

$$S'(\tau') = \left[4 - \tau' + \frac{1}{\pi}\sin\pi\tau'\right] \quad , \quad 2 \leq \tau' \leq 3 \tag{6}$$

$$S'(\tau') = \left[4 - \tau' - \frac{1}{\pi}\sin\pi\tau'\right] \quad , \quad 3 \leq \tau' \leq 4 \tag{7}$$

Also,

$$\tau' = \frac{t'}{T'} \tag{8}$$

where t' is the time in seconds during the obstacle avoidance portion of the trajectory and T' is the total time during this portion, set here at $T' = 4$.

Initially, the preliminary trajectory is designed in the joint space without considering any obstacles and the joint trajectory is broken into small time steps. Then, the Cartesian position of the end-effector is calculated in every time step by solving the forward displacement problem. Thereafter, the distance between the end-effector and the obstacles is calculated and a collision is verified. If a potential collision is detected, the joint trajectory equation changes as follows:

$$\theta_j = \theta_j^I + (\theta_j^F - \theta_j^I)(S(\tau) + \rho S'(\tau')) \tag{9}$$

where ρ is a constant that determines the magnitude of the added joint displacement and is chosen considering the dimensions of the obstacles in the manipulator's path. In [10], the value of ρ is chosen based on experimentation and its values are the same for all the path generator equations in joint space.

This modified trajectory is applied to one or more joints. Note that the method is designed to have the trajectory return to the original pre-designed trajectory after avoiding the obstacles. This method avoids any abrupt changes in displacement, velocity and acceleration in the joint space during the change of the pre-designed path due to the obstacle avoidance sub-task.

4 Path Length Optimization

The obstacle avoidance algorithm proposed in [10] guarantees smooth changes in acceleration while the pre-designed trajectory is changing to avoid obstacles and that is maintained through equation (9). However, the pre-designed trajectory of each of the joints may not need to be altered since the motion of the manipulator is a function of the displacement of all joints. Thus, it may not

be necessary to add the same modifier term $\rho S'(\tau')$ to the path generator equations of all joints. Instead, it is proposed to independently set ρ_j as the amplitude for the modifier function for joint j thus turning equation (9) into

$$\theta_j = \theta_j^I + (\theta_j^F - \theta_j^I)(S(\tau) + \rho_j S'(\tau')) \tag{10}$$

where the values of ρ_j are chosen carefully in a way that the total traversed path of the end-effector remains as short as possible because reducing the path length in Cartesian space results in less displacement in joint space and consequently less energy consumption.

Therefore, the proposed algorithm considers minimizing the end-effector path length in the Cartesian space while imposing constraints on the end-effector path. These two problems may be solved by formulating a constrained optimization problem. The search variables in this case are all amplitudes ρ_j while the objective function to be minimized is defined as the end-effector traversed path during the obstacle avoidance section. The constraints are established as to maintain the robot's end-effector at least at a pre-defined safe distance away from the obstacles in the environment. In other words, the amplitude ρ_j in equation (10) changes for each joint j to maintain the minimum traversed path in Cartesian space. The objective function of the optimization algorithm is written as follow:

$$\begin{aligned} \min \quad & p - p_0 \\ \text{s.t.} \quad & d \geq a \end{aligned} \tag{11}$$

where p_0 is the path length of the pre-designed path when no obstacle is present while p is the length of the current path. Note that p_0 is fixed once the path is given but p is a function of the design variables θ_i and ρ_j. Also, d is used to quantify the shortest distance between the end effector and the object in the environment. Of course, d is a function of the design variables and is to be maintained with a value greater than a used-defined threshold a.

The proposed method is applied for a 6-DOF serial manipulator ($n = 6$) in the next section and the results are compared to those from the method proposed in [10].

5 Numerical Example

A serial MELFA RV-1A manipulator, whose joints are all rotational, is presented in Figure 1 and is used to demonstrate the results of the proposed algorithm. Since the MELFA RV-1A has 6-DOF while the task is only considered to be 3-DOF in Cartesian space (the orientation of the end effector has not been considered in the simulation while all the six joints of the manipulator maintain the motion of the end-effector), it is considered as a redundant manipulator for the task. The manipulator's geometric parameters are established using Craig's notation [11] for the Denavit and Hartenberg (DH) parameters (Table 1) from which the forward displacement solution can be obtained . A smooth path is designed between arbitrary initial and final positions using equation (2) for $T = 15$ s. The Cartesian position of an obstacle which can be detected via cameras or other sensors is presented to the trajectory planning algorithm. Then, the distance between the end-effector and the obstacle is measured in every time iteration to determine the hazardous region. The path generator equation changes to equation (10) if the robot tip gets close to the obstacle.

Table 1: DH parameters [11] of the MELFA RV-1A 6 DOF serial manipulator.

i	α_{i-1} (rad)	a_{i-1} (mm)	d_i (mm)	θ_i
1	π	0	150	θ_1
2	0	250	0	θ_2
3	π	90	43	θ_3
4	π	0	117	θ_4
5	π	0	72	θ_5
6	0	0	90	θ_6

Figure 1: MELFA RV-1A [13]

Thereafter, the modified path planning algorithm designs a new path which avoids any collision between the robot tip and obstacles. Also, equation (10) returns to equation (2) once the end-effector passes the obstacle ($S'(\tau') = 0$). As mentioned earlier, the modifier function ($S'(\tau')$) range remains constant ($T' = 4$) and only the amplitudes ρ_j changes to achieve a path which has the minimum length during the obstacle avoidance algorithm.

In the current Matlab implementation, a constrained nonlinear multi-variable optimization algorithm native to Matlab's optimization toolbox is used (namely, function *fmincon*). This function uses sequential quadratic programming with Hessian updates [12]. The objective function sums the distance between the end-effector position at $time = t$ relative to its position at $time = t - 1$ for the entire trajectory. The optimization problem is constrained to remain away from the obstacle. For simplicity, in the present case, the obstacles are ellipsoids represented only by their major semi-axis.

Figure 2 illustrates the obstacles and the initial trajectory in Cartesian space, denoted as pre-designed, when obstacle avoidance is not considered. The optimized and the non-optimized paths are also shown. The obstacle avoidance algorithm is capable of avoiding collisions both with and without implementing the optimization method. However, the end-effector traversed distance in the Cartesian space is significantly different when the optimization algorithm is implemented. The end-effector path length with the obstacle avoidance algorithm is 560 mm when no amplitude optimization is performed while this length decreases down to 373 mm when the proposed optimization algorithm is applied (the objective function is constrained to keep the end-effector away from the obstacles by at least 100 mm).

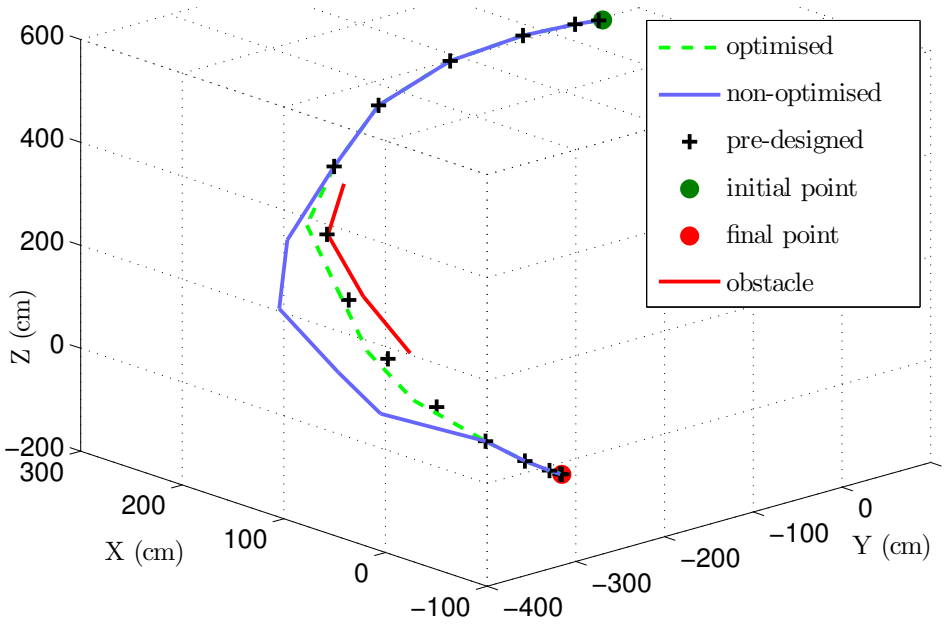

Figure 2: End-effector trajectory in the Cartesian space.

The changes in acceleration are shown in Figure 3. It is clear from there that in either case, the joints' acceleration change smoothly and continuously during obstacle avoidance. The displacements for all six joints of the manipulator is shown in Figure 4 where the joint displacements are shown to be smooth and continuous during obstacle avoidance. It can also be noted that the joint displacement decreases noticeably for most joints while the optimized method is implemented. As another indication of the deviation from the original path, Figure 5 shows the difference between the end effector position and the two collision free paths (optimized and non-optimised paths) and the pre-designed path.

6 CONCLUSION

A simple and efficient joint-space method for obstacle avoidance was proposed. This method yields smooth and continuous changes in actuators displacement, velocity and acceleration while changing the pre-designed trajectory for collision avoidance. By providing trajectories with smooth and continuous joint acceleration profiles, the actuator life time is maximized as infinite jerk spikes are eliminated. Finally, the end-effector path length has been minimized which decreases the the joints displacements as well as energy consumption.

In future work, the authors plan to consider collisions between the manipulator links and the obstacle. The constraint in equation (12) would have to take into account the distance between the end-effector and the obstacle, as well as the shortest distance between each link and the obstacle. Moreover, additional constrains could be added to ensure that the trajectory avoids certain types of singularities.

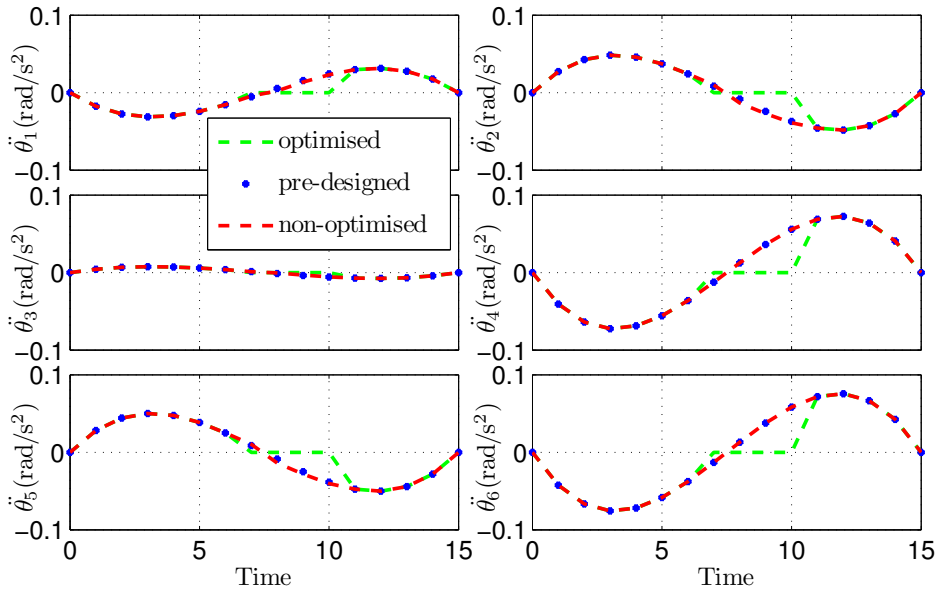

Figure 3: Joint accelerations.

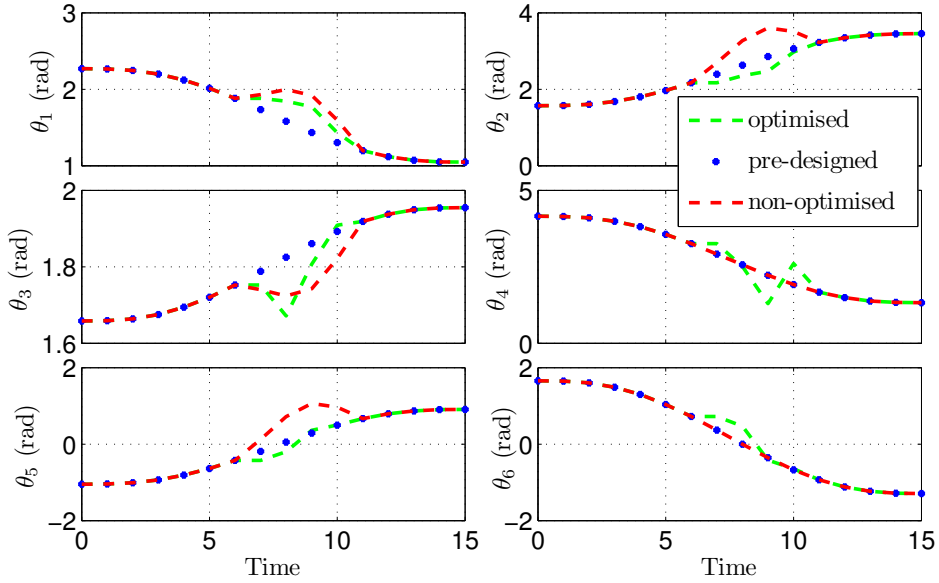

Figure 4: Joint displacements.

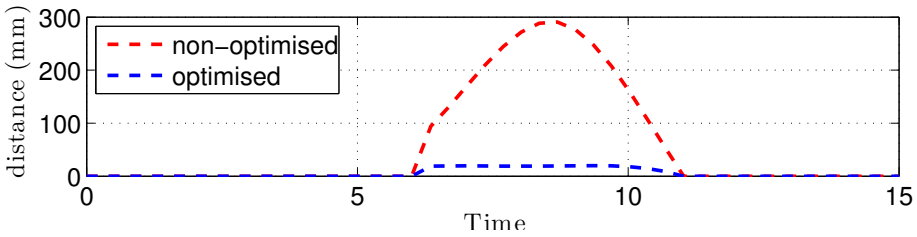

Figure 5: Distance between pre-designed path and actual position.

7 Acknowledgement

The authors of the paper would like to thank the Natural Science and Engineering Research Council of Canada (NSERC) for its support in funding this research.

REFERENCES

[1] Ambike, S., and Schmiedeler, J. P., A Methodology for Implementing the Curative Theory approach to path tracking with planar Robots, Mechanism and Machine Theory, **43**:1225-1235, 2008.

[2] Dash, K. A., Chen, I., Yeo, S. H., and Yang, G., Workspace Generation and Planning Singularity Free Path for Parallel Manipulators, Mechanism and Machine Theory, **40**:776-805, 2005.

[3] Zhou, H., Chen, I., and Ting, K., Path Generation with Singularity Avoidance for Five-bar Slider-crank Parallel Manipulators, Mechanism and Machine Theory, **40**:371-384, 2005.

[4] Bhattacharya, S., Hatwal, H., and Gosh, A., Comparison of an exact and an Approximate Method of Singularity Avoidance in Platform Type Parallel Manipulators, Mechanism and Machine Theory,**33**(7):965-974, 1998.

[5] Dasgupta, B., and Mruthyunjaya, T. S., Singularity Free Path Planning for Stewart Platform Manipulators, Mechanism and Machine Theory, **33**(6):711-725, 1998.

[6] Merlet, J.-P., Trajectory Verification in the Workspace of Parallel Manipulators, International Journal of Robotics Research, **13**(4):326-333, 1994.

[7] Jacobs, P., and Canny, J., Planning Smooth Paths for Mobile Robots, Proc. of the IEEE Int. Conf. on Robotics and Automation, 2-7, April 1989.

[8] Tanner, H., and Kyriakopoulos, K., Nonholonomic Motion Planning for Mobile Manipulators, Proc. of the IEEE Int. Conf. on Robotics and Automation, 1233-1238, April 2000.

[9] Angeles, J., Rojas, A.A. and Lopez-Cajun, C. S., Trajectory Planning in Robotic Continuous-Path Application, IEEE Journal of Robotics and Automation, **4**(4), 380-385, August 1988.

[10] Parsa, S. S., Daniali, H. R. M., and Ghaderi, R., Optimization of Parallel Manipulator Trajectory for Obstacle and Singularity Avoidances Based on Neural Network, International Journal of Advanced Manufacturing Technology, **51**(5-8), 811-816, 2010.

[11] Craig, J. J., *Introduction to Robotics: Mechanics and Control*, Prentice Hall, 3rd ed, 2004.

[12] Byrd, R.H., Gilbert, J. C. and Nocedal, J, A trust region method based on interior point techniques for nonlinear programming, Mathematical Programming, **89**(1), 149-185, November 2000.

[13] Mitsubishi, RV Series, Robots. www.mitsubishi-automation.com, Last accessed: April 16, 2011.