Available online at www.ijapie.org

International Journal of Advanced Production and Industrial Engineering

IJAPIE-SI-RA 802 (2017) 07–11

Special Issue on

ROBOTICS & AUTOMATION

Trajectory Generation Techniques-A Study

Gaganpreet Kaur[1], A. K. Madan[2]

(Delhi Technological University, Delhi, India)
Email: gaganmaotra43@gmail.com

Abstract : *In past much work has been done concerning trajectory generation for manipulators. In this paper, we present comparative analysis of different trajectory generation techniques.*

Keywords: Trajectory, Splines, Algorithm, Cartesian path, Singularity.

INTRODUCTION

In the study of robotics, the connection between the field of study and ourselves is unusually obvious" [Craig,(2005)]. It is for this reason, possibly, that the robotics field interests many of us. Robotics tries to mimic the behavior of human function by the use of revolute joints, sensors, actuators, controllers and computers. There is much research being pursued in different fields of the robotics

Literature survey: A large amount of research has been reported regarding trajectory planning for redundant degree of motion freedom robot manipulators [Whitney (1969), Luh (1985) et.al]. Most of them are based on the calculation of inverse kinematics employing inverse of the Jacobian matrix. Borrow (1988) proposed trajectory planning using the minimal-time criterion was proposed under the B-Spline assumption of the Cartesian path. Sakamoto (1994) proposed the trajectory in the joint space is modeled as a B-Spline curve, and the performance index is integrated in a straightforward manner through the desired trajectory of the end-effectors. Genetic algorithms(GAs) was applied by Davidor (1991) for trajectory generation to pre-defined end-effectors robot paths by searching the inverse kinematics solutions A new method for time-optimal motion planning based on GA was made, which incorporates dynamics constraints, control constraint and kinematics constraints of the robotic manipulator [Yun (1996)et.al].

The trajectory planning is carried out in the joint space and knots connecting through cubic splines the path is represented. Classical optimization techniques, like dynamic programming, fail to be applicable for applications, in particular for real time trajectory planning of manipulators, because of their high complexity. Pires and Machado (2001) proposed a method which optimizes the required manipulating trajectories androbot structure. They described how an manipulator minimizes both the ripple and path trajectory length in the time evolution, without colliding with the obstacles in the workspace. An algorithm containing a GA and a search for pattern is introduced to design the best point-to-point trajectory for a planar 3-DOF manipulator. Rana and Zalzala (1996) described a method to design a near time-optimal, collision-free motion in the case of multi-arm manipulators.

In robotics, one of the major problem of research is to build autonomous, intelligent robots which have the abilityto plan a collision-free path. Roy (2003) described a combined GA and fuzzy logic techniques to solve the trajectory planning of a two-link manipulator. In the proposed method, GAs use optimal tools to find locations along the obstacle-free way and fuzzy logic controller is used to find obstacle-free directions Using Disjunctive Programming, a new algorithm is found for trajectory planning with obstacles for a 2-DOF manipulator [Blackmore (2006)].

Trajectory Generation techniques : Motion control is the most difficult task in robotics. To find the optimal path is the most difficult problem. In this paper we will discuss different methods of trajectory generation in robotics. Basically trajectory generation depends upon the kinematics, the position and velocity of the arm.
Main objective is that we should provide the above parameters and the end effectors will trace the optimal trajectory. But for trajectory generation what technique we should use is a big question so we will be discussing different methods of it and which one will yield the optimal results.

Assumptions in trajectory generation:
1. The trajectory should be specified relative to the stationary frame. We should allow the generalization of moving station frames without significant problems.

2. The trajectory should be smooth, i.e., the position and its first derivate should be smooth. This reduces wear on joint motors and impulsive forces applied to the payload.
3. A trajectory should sat isfy the temporary requirements of the task.
Joint space trajectory schemes In this section, we will discuss trajectory generation methods in which the paths are described in terms of joint angles. The joint angles are generated using the inverse kinematics of the manipulator from the user-defined
Cartesian coordinates. Joint space schemes are usually easy to compute and there is no problem with singularities.
Cubic Spline approach
A common way of causing a manipulator to move from point to point in a smooth controlled fashion is to cause a joint to move as specified by a smooth function of time. Commonly, all joints start and end their motion at the same time, so that the manipulator appears to be coordinated. Trajectory generation help us to compute these motion functions. The trajectory of a manipulator as motions of the tool frame with respect to the stationary frame will be considered so that it can separate the motion descriptions from any particular robot. This helps in the flexibility of different manipulators for path description.

The major problem is to move the tool frame from its current Cartesian position to the destined position, where the motion involves both a change in position and orientation. Usually it wouldbe essential to specify the motion in much more detail than by simply specifying the desired destined position. Knots in the trajectory path help us to create a sequence.
Along with spatial constraints on the motion, the user should specify the time elapsed between knots in the description of the path. The trajectory knots are to be spaced at regular intervals of time.

The basic requirement for trajectorygeneration is that the path should be smooth function that is continuous and has a continuous first derivative. If the generated trajectory is rough and jerky, then it causes wear on the mechanism and cause vibrations by exciting resonances in the manipulator.
Cubic spline functions are the most important spline functions.The reason behind it is that they are smooth functions and, when used for interpolation, they do not have the oscillatory behaviour which is characteristic of high-degree polynomial interpolation.
The univariate or bivariate polynomials have been mostly used for the mathematical formation of splines. A cubic spline function is made by joining various univariate or bivariate cubic polynomials.
The trajectories for the two-link manipulator are as shown in Figure 1. The trajectory time is defined by the user and the knots are at regular intervals o time.
Consider the trajectory time,
$T_i \le t_0 < t_1 < \ldots t_n \le T$
For a single joint we have
$\Theta(t_i) = \Theta_i$ where i=1, 2, 3….n
$\Theta(t_i) = a_i + b_i t + c_i t^2 + d_i t^3$
$t_{i-1} \le t \le t_i$
i=1, 2, 3….n
where (n – 1) is the number of knots between the initial and final knot positions. There are n cubic polynomials to be found, each having four unknown coefficients. Thus, the set of equations to be solved involves 4n unknown coefficients. The ith spline will be evaluated over an interval starting t=ti and ending t=ti+1 where i =0, 1, 2….n-1
To obtain the coefficient there should be 4n constraints. The constraints are given in Equation below and the continuity

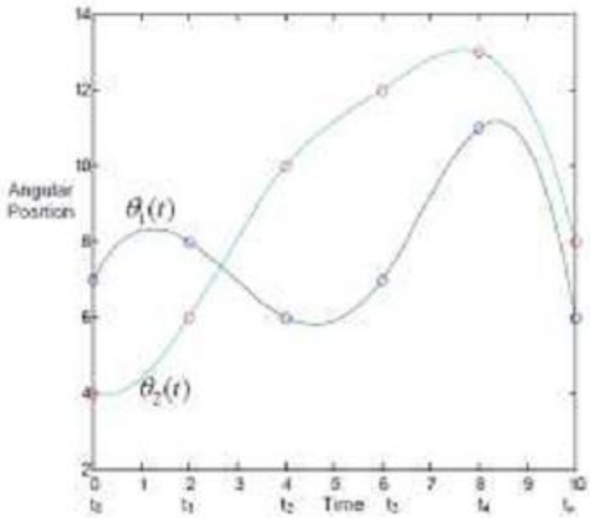

Figure1. Joint Trajectories of two link

restrictions
$\theta^j(t_i^+) = \theta^j(t_i^-)$
i=1……(n-1) j=0,1,2
Together it gives n+1+3(n-1)=4n-2 constraints, as compared to 4n unknowns.
For interpolation problem, there are two more degrees of freedom in choosing the Coefficients of above equation. The manipulator is considered to be at zero velocity at start and end positions.
so $\theta_1(t_o)=0$ and $\theta_n(t_n)=0$
Solving the 4n constraint linear equations, we get the cubic spline coefficients which result in describing the trajectory of the joint passing through the specified knots

Linear function with parabolic blends:
A simpler interpolation scheme than the polynomial approach is linear interpolation. That is, there is linear interpolation between the initial and final joint position as shown in Figure2. Although the motion of each joint is linear, the end-effectors in general does not move in a straight line in space. The main problem is that at via points, the velocity and acceleration will be discontinuous. A good solution would be to add a “parabolic blend" section at the via point to interface the two interpolating curves. During the blend portion of the curve, constant acceleration is choosen to change the velocity smoothly. Figure 3 shows a simple path constructed in this way.

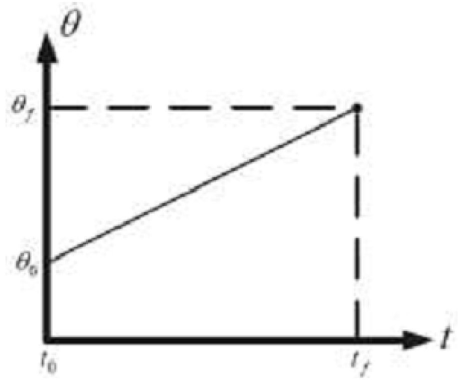

Figure3. Linear segments with parabolic

Depending on the value of acceleration chosen, and on the change in velocity required between adjacent linear sections, the blend region will extend further or less into the linear region.

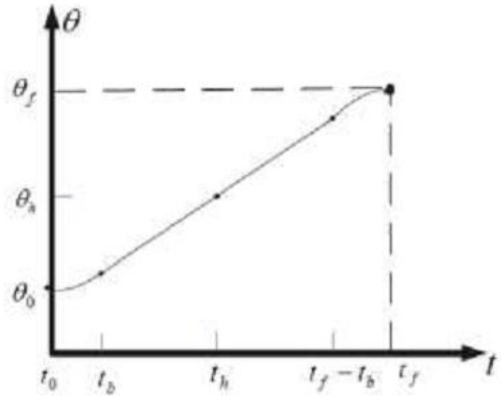

Figure3. Linear segments with parabolic blends

There can be another case of linear interpolation where trapezoidal velocity profile can be chosen to produce a joint trajectory.

Cartesian space trajectory schemes:

In the previous trajectory generation techniques the paths computed in the joint space, is via the start and end points even they are specified in a Cartesian frame. But the path followed by the manipulator was not a straight line connecting the start and end points. But it would be some complex shape that depends on the trajectory approach.

Cartesian trajectories give the actual motion of the end-effectors of the robot. However, Cartesian motion of the robot does not map trivially to a trajectory in joint space. The trajectory made from a Cartesian path is generally more complex in joint space than by direct interpolation which wil l result in high actuation requirements on all joints.

In Cartesian trajectories the motion of the end-effectors is smooth and natural. That is why there are less forces like inertia and gyroscopic disturbances on the manipulator because of the load carried by the end-effectors.

By repeated application of the inverse kinematics at every point in Cartesian paths joint motion trajectory is obtained. Since trajectories are not generated in joint space, so it should be taken notice that the path lies in the reachable work space and does not pass through singularities.

Path planning algorithm:

Various techniques for generating Cartesian paths are proposed in the literature .The path planning algorithm plans the profile with specified current position, initial velocity, and distance to travel. In case of a time-based planner, the time for the move (Te) is specified instead of the target velocity.

The initial conditions for the interpolator as used by the algorithm are denoted as follows:

Te = Time for the move
Dtg = Distance to travel
Vi = Initial velocity
Acc = Specified acceleration
Dec = Specified deceleration
Vmax = maximum target velocity for the interpolator.

There are the two profiles types based on the initial conditions, Vmax, and total time, Te. The algorithm determines which profile type is needed.

Profile Type – A

cart1 = {X1, Y1}; cart2 = {X2, Y2};

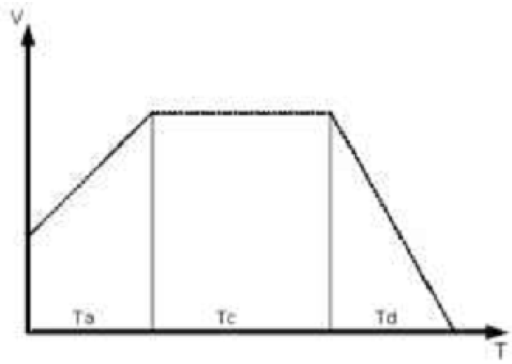

Figure 4

Distance travelled for Type-A = Dtg = norm (cart1 – cart2) = Accel Distance + Coast Distance + Decel Distance

$$\text{Accel Distance}=DTa=\frac{V_{MAX}^2 - V_i^2}{2\,Acc}$$

$$\text{Decel Distance}=Dtd=\frac{v_{max}^2}{2Dec}$$

Coast Distance=Dtg-Dta-Dtd

$$Ta=\frac{V_{max}-V_i}{Acc};\ Td=\frac{Vmax}{Dec}\ ;\ Tc=\frac{DTc}{Vmax}$$

Te = Total time = Ta + Tc +Td;

Profile Type – B

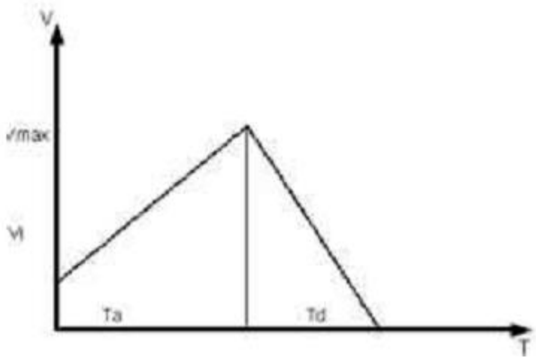

Figure 5

Distance traveled for Type-B =
Dtg = norm(cart1 – cart2) = Accel Distance + Decel Distance

Accel Distance=DTa= $\frac{V_{MAX}^2 - V_i^2}{2\,Acc}$

Decel Distance =Dtd = $\frac{v_{max}^2}{2Dec}$

$$Ta=\frac{V_{max}-Vi}{Acc};\ Td=\frac{Vmax}{Dec}$$

$$V_{MAX}=\sqrt[2]{\frac{2DTG}{\frac{1}{Acc}+\frac{1}{Dec}}}$$

The inverse kinematics routine transforms the Cartesian information into joint angles and at runtime the path generator routine constructs the trajectory at the path-update rate which is fed to the manipulator's control system Problems Cartesian paths, even though the initial point and the final point are in the reachable workspace, but not all points which are on the straight line between these two points are in the workspace. For instance, consider the two link manipulator as shown in Figure 4. In this case, link1 is greater than link2, so the workspace contains an inner radius in the middle whose radius is the difference between the link lengths. If we draw a straight line starting from the initial point A to a goal point B and attempt to make a Cartesian move, the intermediate points will not be reachable. In that case we need to employ joint space schemes, which are an advantage over Cartesian schemes.

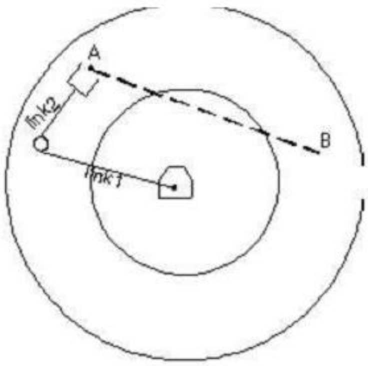

Figure 6.Two link manipulator trying to move in path A to B.

Joint velocities near singularity It is impossible to limit the joint velocities that yield the desired Cartesian velocity for the end-effectors. If, for example, a manipulator is following a Cartesian straight line path and approaches a singular configuration of the mechanism, one or more joint velocities will increase towards infinity.

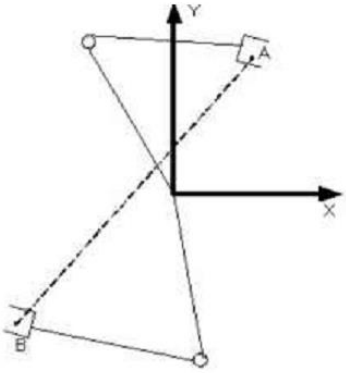

Figure 7. High joint velocities near singularity

As an example, Figure shows a two-link manipulator with equal link lengths moving along a path from point A to point B. The desired path is to move the tool tip along this straight line maintaining a constant linear velocity. All points along the path would be reachable, but as the robot gets near to the singularity, which is the origin in this case, the velocity of the joints becomes very high. This problem is also observed when the manipulator is getting near to the fully stretched singularity condition. There might be cases where one is required to follow a Cartesian path which approaches the singularity condition. One solution would be to program the system in such a way that the move is completed in three separate moves.

The first programmed move will be a Cartesian path from the initial point to a point very close to the origin. Then a very small linear joint move is programmed, followed by the Cartesian path to the end point.

Lefty- Righty solutions with cartesian paths

For a single Cartesian point there, are multiple ways of approaching the point. In case of the two-link manipulator, it would be a right arm solution and a left arm solution.

For example, if we want to program a Cartesian path from point A to point B, as shown in Figure 6, we can see that by the time it reaches the final point, the approach solution is changed from a right arm configuration to a left arm configuration.

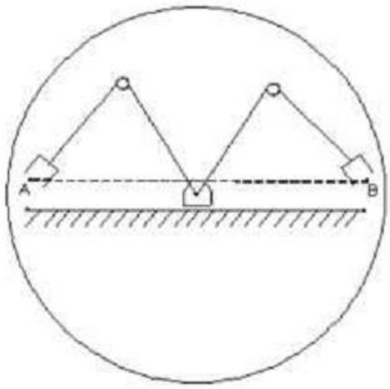

Figure 8.Start and goal reachable in different solutions.

CONCLUSION :

Trajectory generation at update period For all the above trajectory schemes, the final trajectory path is a set of data for each segment of the trajectory. At run time, the interpolator routine generates the trajectory position, velocity and acceleration, and feeds the information to the manipulators control system at the path update period.

In the case of cubic splines, the path generator simply computes as t is advanced. When the end of one segment is reached, a new set of cubic coefficients is recalled, t is set back to zero, and the generation continues. In the case of linear splines with parabolic blends, the value of time, t, is checked on each update to determine whether we are currently in the linear or blend portion of the segment.

Because a continuous correspondence is made between a path shape described in Cartesian space and joint positions, Cartesian paths are prone to various problems relating to workspace and singularities.

So a cubic spline approach for optimal trajectory generation is employed and compared against linear interpolation. The amount of acceleration that the manipulator is capable of at any instant of time is a function of the dynamics of the arm and the actuator limits. Further we can select trajectory generation schemes on the basis of dynamics involved in manipulators.

REFERENCES

[1] John J. Craig, "Introduction To Robotics - Mechanics and Control" 3rd edi t io n, Pearson Prentice Hall, 2005

[2] D. E. Whitney, "Resolved motion rate control of manipulators and human Prostheses," IEEE Transactions on systems, Man and Cybernetics, Vol. MMS-10, pp. 47-53, 1969

[3] J. Y. S. Luh, Y. L. Gu, "Industrial Robots with seven Joints," Proc. IEEE International Conference on Robotics and Automation, Mar. 1985

[4] J. E .Borrow, "Optimal Robot path Planning Using the Minimum-Time Criterion," EEE Journal of Robotics and Automation, Vol. 4, pp. 443-450, August 1988

[5] K. Sakamanto, A. Kawamure, "Trajectory Generation for Redundant Robot Manipulator by Variational Approach," Advance motion control-San Franciso, pp. 378-385, March 1994

[6] Y. Davidor, "Genetic Algorithms and Robotics, a Heuristic Strategy for Optimization," World Scientific,1991.

[7] W. M. Yun, Y. G. Xi, "Optimum motion planning in joint space for robots using genetic algorithm," IEEE on Robotics and Automation Systems, Vol. 18, pp. 373-393, 1996.

[8] A. Rana, A. Zalzala, "An Evolutionary Planner for Near Time-Optimal Collision-Free Motion of Multi-Arm Robotic Manipulators", Int. Conf. Control, Vol. 1, pp. 29-35, 1996.

[9] E.J. Solteiro Pires, J. A Tenreiro Machado, P.B. de Mour Oliveira, "An evolutionary Approach to Robot Structure and Trajectory Optimization,"ICAR 01-10th International Conference on Advanced Robotics, Budapest, Hungary, pp. 333-338, 22-25/Aug/2001.

[10] A. A. Ata, R. T. Myo, "Optimal Point-to-Point Trajectory Tracking of Redundant Manipulators using Generalized Pattern Search," International Journal of Advanced Robotic Systems, Volume 2, Number 3, pp.239-244, 2005.

[11] S. S. Roy, "A Genetic-Fuzzy Approach for Optimal Path-planning of a robotic Manipulator among Static Obstacles," The Institution of Engineers (India), Volume 84, pp. 15-22, May 2003.

[12] L. Blackmore, B. Williams, Optimal Manipulator Path Planning with Obstacles using Disjunctive Programming," American Control Conference, 2006.

[13] W. J. Book, "Recursive Langrangian dynamics of flexible manipulator arms," International Journal of Robotics Research, pp. 87-101, 1984

[14] K. Chang, O. Khatib, "Manipulator Control at Kinematic Singularities: A Dynamically Consistent Strategy," Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems, Pittsburgh, Vol. 3, pp. 84-88, August1995.

[15] C. DeBoor, "A Practical Guide to Splines," Springer-Verlag, New York, 1978

[16] K. Atkinson, "An Introduction to Numerical Analysis," Wiley, New York, 1989.

[17] R. P. Paul, H. Zong, "Robot Motion Trajectory Specification and Generation,"2nd International Symposium on Robotics Research, Kyoto, Japan, August 1984.

[18] R. Taylor, "Planning and Execution of Straight Line Manipulator Trajectories," in Robot Motion, Brady et al., Editors, MIT Press, Cambridge, MA 1983.

[19] K. S. Fu, R.C. Gonzalez, C. S. G. Lee, Robotics Tutorials, IEEE Press, 1990.

[20] J. Y. S. Luh, M. W. Walker, R. Paul, "On-Line Computational Scheme for Mechanical Manipulators," Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control, pp. 69-76, 1980.