Path planning by hybrid PSO-Splines algorithm

Paulo Salgado
psal@utad.pt

ECT- Departamento de Engenharias,
Universidade de Trás-os-Montes e Alto Douro, Vila Real, Portugal

Abstract

Trajectory planning is an NP-hard problem, and the computational effort of planning methods is usually high. In this paper is presented an algorithm for planning and optimizes trajectories. The trajectory is defined via Spline Cubic Polynomial interpolation, which represents continuous particles in a continuous search space. Therefore, to find the quasi-optimal trajectory, an adapted Particle Swarm Optimization algorithm is used, where particles are continuous spline functions. When applied to robots, this algorithm generates smooth motion trajectories, with two times continuously differentiable curves, and avoiding obstacles placed in the workspace. Thus, it can be used for autonomous robot path planning or transportation problems. It methods is also appropriate to contours image segmentation task. It was tested by hard 2D and 3D problems and the results demonstrated the effectiveness and performance of the algorithm.

1 Introduction

The scientific and industrial community have given much effort on creating optimization algorithms to solve global path planning problems. Also, there is a growing interest on path planning to be use by mobile robots, cars with autonomous driving, drones and industrial machines for many proposals [1-2]. This interest is now extended in contour task (segmentation) of digital image or to support analysis methods [3-4].

Path planning (PP) is only possible when the environment map is known. For instance, the robot's movement involves moving along a trajectory, starting from a specific point and finishing in an endpoint, passing through a sequence of points on its way and avoiding collision with other objects in the same workspace. Additionally, a robot's trajectory includes the motion of the robot with respect to time that is constrained by kinematic limits (e.g., joint velocity limit) and dynamic limits (e.g., torque constraint) of the robot joints and motors.

Hence, the goal is to plan and coordinate the motion of the robot axis. This is possible by fulfilling all constraints in order the robot passes all waypoints without any collisions and finally reaching its destination. The basic requirements of a good trajectory are the combination of smooth motion, a short and safety path. If these objectives are fulfilled, robots can efficiently improve its trajectory performance, namely by reducing the time of travel and the energy consumption [5].

Many trajectory-generation methods were proposed in the literature [6-7], which created motion trajectories for the robots considering several simultaneous criteria and constraints such as travelling distance, smoothness, security and feasibility [8], which it is almost a NP-hard problem. For the majority of the problems, it doesn´t exist an explicit solution or a deterministic method to solve them in an optimal and global way, taking into account all the constraints. To overcome these difficulties, the problem is usually simplified and therefore, the quality of the path solution is reduced. For many of these methods, a set of points with specific constraints is given and a path is generated from the combination of straight lines and circular arcs [9]. However, there is a curvature discontinuity at the straight line and at the circular arc joint. To cope with this problem, many researchers have modelled robot trajectories as piecewise quadratic or cubic Bézier curves [10]. Cubic polynomials splines have also been widely used as single curves to generate two times continuously differentiable curvature (trajectories), C^2, [11-13].

As I mentioned, in this paper I use the PSO algorithm to find a quasi-optimal trajectory through a feasible path in a complex environment, using a baseline smooth path based on cubic splines. This work is a step ahead of the algorithms proposed in [14], now with the improvement of the attraction/repulsive force of the Spline particles of the PSO algorithms. This method does not use traditional particle entities. Instead, they were replaced by continuous functions, namely by cubic splines, obeying certain C^2 continuity requirements throughout their domain and interpolating waypoints [15]. It is able to (iteratively) refine the path and thus find an efficient, collision free path in real time through an unstructured environment. This method is validated, and its performance evaluated through a set of simulations of hard and complex problems, with a great number of circular or spherical obstacles. The algorithm demonstrates a high success rate for all of the tested environments.

The rest of this paper is organized as it follows. Section 2 introduces the problem formulation for path planning. Section 3 introduces the PSO-Cubic Spline algorithm, PSO-CS, an optimization method in the space of continuous spline functions, and Section 4 presents experiments and result analysis. Finally, Section 7 summarizes the whole paper and presents the main conclusion.

2 Trajectory planning problem

Let a workspace W with n obstacles O_o , $o=1,\cdots,n$, and a set of trajectories $T_k=\left\{\left(t_j, X_j^{(k)}\right), j=0,\cdots,m+1\right\}$ where m are the number of waypoints with coordinates $X_j^{(k)} \in W$, the desired location of the robot at time t_j , specified in task space. It is assumed here that the positions of the starting and ending waypoints are provided.

For a given set of waypoints there is a unique piecewise-cubic trajectory, $S(t)$, that passes thr $y_j(t)=S_{y,j}(t)$ ough the points and satisfies a certain smoothness criterion. Specifically, let the Spline trajectory in $W \subset \boldsymbol{R}^3$ between times t_{j-1} and t_j as a cubic functions, one for each coordinate system: $x_j(t)=S_{xj}(t)$, and $z_j(t)=S_{z,j}(t)$, where S is a cubic polygon, i.e., $S_{r,j}=a_{rj}\Delta t_j^3+b_{rj}\Delta t_j^2+c_{rj}\Delta t_j+r_{j-1}\left(t_{j-1}\right)$ for $r\in\{x,y,z\}$. Founded a set of sequential waypoints, there exists a unique set of coefficients $\left\{\left(a_{rj},b_{rj},c_{rj}\right)_{r=\{x,y\}}\in\boldsymbol{R}^3\right\}_{j=1,\cdots,m}$, such that the resulting trajectory passes through the waypoints and has continuous C^2 profiles in the complete path. The waypoints are the visible solution of the optimization process performed by the PSO algorithm. However, in their evolutionary strategy, it takes into account not directly these points, but the complete path trajectory curve (spline) by evaluating its performance in the generated path. Truly, the PSO-CS, is an optimization method in the space of continuous spline functions.

3 Spline PSO algorithm

Particle Swarm Optimization, PSO, is inspired by the social behaviour of some biological organisms, especially the ability of some animal species to locate a desirable position in a given area. There are examples of this social behaviour on flock of birds and shoal of fish. This method is one of the optimization methods developed for finding a global optimal of some nonlinear function [18].

This method applies the approach of problem solving by a population of candidates solutions. Each solution consists on a set of parameters and represents a point in multidimensional space. The solution is called "particle" and the group of particles (population) is called "swarm". These particles move inside the search-space according to a few simple formula, they are guided by their own best known position in the search-space as well as the entire swarm's best known position, iteratively trying to improve a candidate solution with regard to a given measure of quality. So, each particle *i* is represented in a D-dimensional by the position vector $\vec{x}_i$ and it has a corresponding instantaneous velocity vector $\vec{v}_i$. It has a memory that tracks a best position of the previous iteration: the particle's optimal position *pbest* and the particle's global optimal position *gbest*. The particles are moving in *W*, under an action of one force that results from random combination of three effects: inertial, attraction of *pbest* and attraction of *gbest*. Under the effect of this force, the speed of particle is update in each iteration as:

$$\vec{v}_i(k+1)=\alpha\vec{v}_i(k)+c_1r_1\left(\vec{p}_{best,i}-\vec{x}_i(k)\right)+c_2r_2\left(\vec{p}_{gbest}-\vec{x}_i(k)\right) \quad (1)$$

where α is an inertia weight parameter, r´s are random numbers drawn from a uniform distribution in interval $[0,1]$, c_1 and c_2 are weights also designed as 'cognitive acceleration coefficient', respectively for local or global best position. The components values of $\vec{v}_i$ is restrict into the interval $[-v_{max}, v_{max}]$. Next, the position update rule (2) is applied:

$$\vec{x}_i(k+1) = \vec{x}_i(k) + \vec{v}_i(k). \quad (2)$$

In this work, the PSO method does not use traditional particle entities. Instead, they were replaced by continuous functions, namely by cubic splines. Cubic Spline that obeys a certain C^2 continuity requirements throughout their domain and interpolating waypoints defines the path. The arcs of Spline are subject to a force of repulsion by obstacle objects. This fourth component of the force that moves the particle in *W*.

Despite the differences, for simplicity of analysis, we are going to use the same designations indistinctly. S_i represents the i^{th} Spline particle, with waypoints X_i in *W*, that it have an instantaneous velocity vector function $\vec{u}_i(t)$. $\vec{U}_{ij} = \vec{u}_i(t_j)$ is the velocity vector of the j^{th} waypoint of spline particle i with travel trajectory time t_j. Under the influence of near objects, the arcs of Spline are subjects to a repulsion force. This force has a higher magnitude value if the arc spline is inside object body and, consequently, a less magnitude value for spline arcs further from the object. It has a null value for distances greater than the distance of safety margin. This tensor or strain force that propagates across Spline curve to their waypoints results in the force R_i. This joins to the other PSO force, which represents another right term of equation (1).

4 Tests and results

I tested the PSO-CS algorithm for robot path planning in ®MATLAB platform in two test-examples, the 1st in two-dimensional space, 2D, and the 2nd in three-dimensional space, 3D. The workspace has a shape like a square/cube with edge length of 100 units. Inside there are 30 circular/spherical obstacles with radius length of 10 units. In the first example, the circles are random placed on workspace while in the 2nd text-example a wall, made by 24 spheres and a hole at its centre that split the workspace in two zones. Three more spheres are randomly placed in each one-sided zones. We assume that the start position of the robot is in origin, $X_0 = [0,0] / X_0 = [0,0,0]$, and the end position is in opposite vertices, i.e $X_{m+1} = [100,100] / X_{m+1} = [100,100,100]$.

PSO-CS uses Spline curves as particles. It adjusts the randomly placed waypoints of Splines population, where the particles are the coordinates of these points, in a discrete optimization process. The spline particles have 5 waypoints. One hundred 'particles' have been used for simulations with 100 iterations. For both environments, the PSO-CS was tested. The best Spline path is recorded at each iteration. At the end of the process, the best Spline trajectory is shown, as well as the final population and the best performance evaluation in each iteration. Figure 1 and Figure 2 show the results, respectively, for the 2D and 3 D test-examples. The red line shown in figures represents the optimum path generated by the algorithms and the filled circles/spheres represent obstacles.

From simulations results, we can conclude that PSO-CS algorithm efficiently finds a collision-free path between the initial and destination points, the global solution with a quasi-optimal performance value.

5 Conclusions

This paper presents an improved version of the PSO-CS algorithm for global path planning. In order to get smoother planned paths, it uses a Cubic-spline smoothing technique. It was tested for robot motion planning problem, which it is treated as an optimization problem. Random obstacles are place on the 2D and 3D workspaces. The PSO-CS, in each iteration, try's to find a feasible Spline curve with the best performance, defined by appropriated waypoints. The result is a smoother planned path, which it is a quasi-optimal trajectory. Experimental results show that it is an excellent method for path planning, generating collision free and smooth trajectories with shorter paths length.

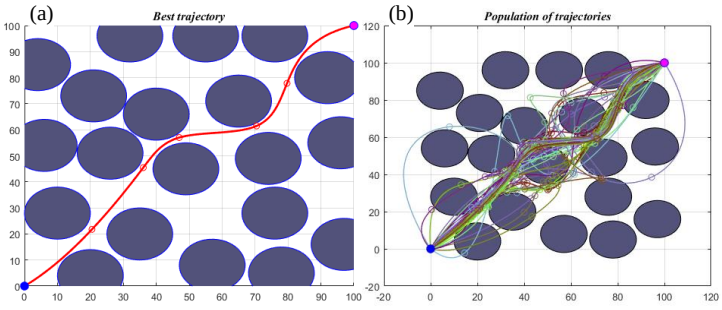

Figure 1: PSO-CS: (a) Best trajectory; (b) Population of splines.

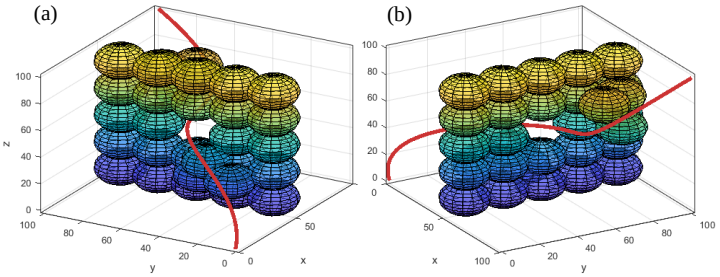

Figure 2: PSO-CS trajectory. View: (a) El. 30º, Ez. -60º and (b) Ez. 60º.

References

[1] Luigi Biagiotti and C. Melchiorri, "Trajectory Planning for Automatic Machines and Robots", Springer; 2008.

[2] Steven M. LaValle, Planning Algorithms, Cambridge Univ., 2006.

[3] R. Delgado-Gonzalo, M. Unser, Spline-based framework for interactive segmentation in biomedical imaging, IRBM, Vol. 34 (3), pp. 235-243, 2013. https://doi.org/10.1016/j.irbm.2013.04.002.

[4] L. Stanberry, J. Besag, Boundary reconstruction in binary images using splines, Pattern Recognition, Vol. 47(2), pp. 634-642, 2014.

[5] Steven M. LaValle, Planning Algorithms, Cambridge Univ., 2006.

[6] Serdar Kucuk, Optimal trajectory generation algorithm for serial and parallel manipulators, Robotics and Computer-Integrated Manufacturing, Vol. 48, pp. 219-232, 2017.

[7] B.K. Patle and at. al., A review: On path planning strategies for navigation of mobile robot, Defence Tech., Vol. 15(4), pp. 582-606, 2019.

[8] Prasenjit Chanak, and at.al., "Obstacle Avoidance Routing Scheme Through Optimal Sink Movement for Home Monitoring and Mobile Robotic Consumer Devices", IEEE Trans. on Consumer Electronics, Vol. 60, No. 4, pp. 596-604, Nov. 2014.

[9] Yi Fang and at. al., Smooth and time-optimal S-curve trajectory planning for automated robots and machines, Mechanism and Machine Theory, Vol. 137, Pages 127-153, 2019.

[10] Christian Scheiderer and at. al., Bézier Curve Based Continuous and Smooth Motion Planning for Self-Learning Industrial Robots, Procedia Manufacturing, Vol. 38, pp. 423-430, 2019.

[11] T. Berglund and at. al., "Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles," IEEE Trans. Autom. Sci. Eng., vol. 7, no. 1, pp. 167–172, Jan. 2010.

[12] M. Elbanhawi and at. al., Continuous path smoothing for car-like robots using b-spline curves, J. Int. Rob. Syst., vol. 80(1), pp. 23–56, 2015.

[13] R. Cowlagi and P. Tsiotras, "Curvature-bounded traversability analysis in motion planning for mobile robots," IEEE Trans. Robot., vol. 30, no. 4, pp. 1011–1019, Aug. 2014.

[14] P. Salgado, and at. al, "Hybrid PSO-cubic spline for autonomous robots optimal trajectory planning," 2017 IEEE 21st Int. Conf. on Intelligent Engineering Systems (INES), Larnaca, pp. 131-136, 2017, doi: 10.1109/INES.2017.8118542.

[15] G. Liu and S. Wu, "A Novel Optimized Trajectory Planning Method via Spline Function Theorems," 2018 IEEE 4th Inf. Tech. and Mech. Eng. Conf. (ITOEC), China, pp. 745-749, 2018.