ISSN: 1813-4890

Path Optimization for Complex Dynamic Systems

Pingping Bing [1,*], Jiaoyun Liu [2] and Liyuan Chu [1]

[1]International Medicine Institute, Changsha Medical University, Changsha, China

[2]School of Information Engineering, Changsha Medical University, Changsha, China

*Corresponding author: bpping@163.com

Abstract

This study focuses on the path optimization problem of the "Bench Dragon" in a dynamic system, constructing a mathematical model based on the equiangular spiral equation and a U-turn space model, and applying a simulated annealing algorithm for optimization calculations. By precisely calculating the movement parameters of the dragon's head and each section of the benches, the U-turn path length is minimized, ensuring no collisions during the system's turning process. Experimental results show that by adjusting the spiral pitch and optimizing the U-turn path, the path length is reduced to a minimum value of 13.7212 meters, enhancing both motion efficiency and system stability. This provides a theoretical reference for path planning and optimization in complex dynamic systems.

Keywords

Simulated annealing algorithm; spiral curve; path optimization.

1. Introduction

The "Bench Dragon" forms a long-chain system with a complex dynamic structure by linking multiple benches end-to-end. As urban spatial structures become more complex and public activity demands increase, this dynamic chain system faces challenges related to spatial utilization efficiency and the precision of motion paths in modern urban environments[1-4]. This study aims to transform the path optimization problem of the "Bench Dragon" into a typical chain motion system optimization problem. We developed a mathematical optimization model based on the spiral equation and U-turn space model. The model is solved using a simulated annealing algorithm to minimize path length, precisely control U-turn areas, and ensure the stability of the system's dynamic behavior.

2. Method

2.1. Path Modeling and Equation Construction

2.1.1. Path Description

To accurately describe the coiling motion of the "Bench Dragon," we use a polar coordinate system to define its path. Assuming the dragon's head coils clockwise along an equiangular spiral with a pitch of p, the path can be represented by a spiral curve. The path equation in polar coordinates can be defined as follows:

$$r(\theta) = r_0 + \frac{p}{2\pi} \cdot \theta \tag{1}$$

ISSN: 1813-4890

Where r(θ) is the radial distance, representing the radius of the spiral as a function of the polar angle θ; r_0 is the initial radius of the spiral; pis the pitch, i.e., the change in radius per full rotation of the spiral, given as p=0.55m, and the spiral is equiangular.

2.1.2. Position Calculation

The "Bench Dragon" consists of multiple benches connected end-to-end. To accurately simulate the motion trajectory of each bench section, we must account for the offset of each section relative to the dragon's head. Assuming the length of each bench is L, the polar angle and radial distance of the n-th bench section are given by:

$$\theta_n(t) = \theta_{\text{head}}(t) + \frac{n \cdot L}{r(\theta)} \tag{2}$$

$$r_n(t) = r(\theta_n(t))$$

This allows us to calculate the polar coordinates for each bench section in sequence. These polar coordinates are then converted to Cartesian coordinates to obtain the position of each bench section at time t.

2.1.3. Velocity Calculation

The velocity of each bench section can be calculated by the displacement between adjacent time steps. The velocity is obtained by integrating the displacement of the bench section over time:

$$v(\Delta t) = l = \int_{\theta_1^0}^{\theta_1^1} \sqrt{[\rho(\theta)]^2 + [p'(\theta)]^2}\, d\theta \tag{3}$$

2.2. Collision Detection Model

The "Bench Dragon" is composed of multiple connected benches, and during motion, each bench section must avoid collisions. To achieve this, we construct a collision detection model using geometric analysis to determine if any bench sections overlap. Specifically, each bench section is approximated as a rectangle, and we calculate the positions of the four corner points of adjacent bench sections to check for overlap. Let the four corner points of the bench be M_n, M_n', N_n, N_n'. Based on the symmetry of the rectangle, we can derive:

$$\begin{aligned}
M_n &: \left(\frac{r(t)\times\cos(\theta(t))}{2} - \frac{w}{2}, \frac{r(t)\times\cos(\theta(t))}{2} + \frac{h}{2}\right)\\
M_n' &: \left(\frac{r(t)\times\cos(\theta(t))}{2} - \frac{w}{2}, \frac{r(t)\times\cos(\theta(t))}{2} - \frac{h}{2}\right)\\
N_n &: \left(\frac{r(t)\times\cos(\theta(t))}{2} + \frac{w}{2}, \frac{r(t)\times\cos(\theta(t))}{2} + \frac{h}{2}\right)\\
N_n' &: \left(\frac{r(t)\times\cos(\theta(t))}{2} + \frac{w}{2}, \frac{r(t)\times\cos(\theta(t))}{2} - \frac{h}{2}\right)
\end{aligned} \tag{4}$$

ISSN: 1813-4890

To detect collisions between adjacent benches, we use the Separating Axis Theorem (SAT). This theorem states that if the projections of two rectangles on any separating axis do not overlap, then the rectangles do not collide. The specific steps are as follows:

Determine separating axes: For each pair of adjacent benches, select their edges as separating axes, and compute the projections on these axes.

Compute projections: Calculate the projections of the four corner points of each bench on the separating axes and check if the projections overlap.

Collision detection: If the projections do not overlap on all separating axes, the benches are not colliding; if at least one separating axis shows overlapping projections, a collision has occurred.

2.3. U-Turn Space Model

2.3.1. Geometric Construction of the U-Turn Space

To enable the "Bench Dragon" to turn smoothly in a limited space, the U-turn space is defined as a circular region with a radius R_{turn}, centered at the spiral's origin. Within this region, the "Bench Dragon" must perform a U-turn from coiling inward to coiling outward, ensuring continuity and smoothness of the path as the dragon's head enters and exits the region.

Geometric definition: The radius of the U-turn space is R_{turn}. For example, R_{turn}=4.5m. This radius is determined based on the length of the benches and the minimum turning radius required during the U-turn.

Boundary conditions: The boundary of the U-turn space is the outer edge of the circle, and the U-turn process must be completed within this area without collisions. Thus, the dragon's head must remain inside the circular region throughout the turning maneuver.

2.3.2. Calculation of Minimum Pitch

To ensure that the "Bench Dragon" can smoothly enter the turnaround space without collision, we need to determine a minimum pitch (p), which is the smallest distance between each coil of the spiral path. This allows the dragon's head to reach the boundary of the turnaround space while maintaining continuity and avoiding collisions. The calculation process is as follows:

Spiral Equation: Assume the spiral equation is $r(\theta) = r_0 + (p / 2\pi) * \theta$, where p is the pitch and r_0 is the initial radius of the spiral. The boundary radius of the turnaround space is defined as R_{turn}, and the condition for the dragon's head to reach this boundary is:

$$r(\theta_{\text{turn}}) = R_{\text{turn}} \tag{5}$$

Polar Angle Calculation: Using the above equation, we can solve for the polar angle θ_{turn} when the dragon's head reaches the boundary of the turnaround space:

$$\theta_{\text{turn}} = \frac{2\pi(R_{\text{turn}} - r_0)}{p} \tag{6}$$

Time Calculation: Assume the speed of the dragon's head is constant v_{head}, the time t_{turn} for the dragon's head to reach the boundary of the turnaround space can be expressed as:

$$t_{\text{turn}} = \frac{\theta_{\text{turn}} \cdot r_0}{v_{\text{head}}} \tag{6}$$

ISSN: 1813-4890

By adjusting the pitch p under the condition of no collision, we can calculate the minimum pitch that allows the dragon's head to reach the boundary of the turnaround space while satisfying the above conditions.

2.3.3. Modeling the S-shaped Turnaround CurveS

In the turnaround space, to achieve a smooth turnaround for the dragon's head, the path is designed as an "S"-shaped curve. This curve is composed of two tangent circular arcs: the first arc has a smaller radius R, and the second arc has a radius double that, i.e., 2R.

Small Arc Construction: Assume the small arc has a radius R and is centered at the center of the turnaround space. Define the start and end points of this arc, which forms the initial part of the turnaround path.

Large Arc Construction: The large arc has a radius of 2R and is tangent to the small arc. This large arc forms the second part of the turnaround path, ensuring that the dragon's head smoothly completes the direction change and continues coiling counterclockwise.

Tangency Condition: To ensure continuity and smoothness of the turnaround curve, the tangency condition is set so that the tangent directions of the two arcs are the same at the tangency point. This prevents abrupt changes in the dragon's head movement when entering and exiting the turnaround area.

2.3.4. Optimization of Turnaround Curve Parameters

To minimize the length of the turnaround path, the parameters of the turnaround curve (such as the small arc radius R) need to be optimized.

Objective Function. The total length of the turnaround curve is defined as the optimization objective, with the function being:

$$L_{\text{turn}} = L_{\text{small}} + L_{\text{large}} \tag{6}$$

Where L_{small} and L_{large} are the lengths of the small and large arcs, respectively.

2.4. Simulated Annealing Algorithm for Optimizing the Turnaround Path Model

2.4.1. Basic Principles of the Simulated Annealing Algorithm

The Simulated Annealing (SA) algorithm is a global optimization method based on the physical annealing process[5-8]. It simulates the cooling process of materials at high temperatures, gradually lowering the system's energy to find the optimal solution. In the "Bench Dragon" path optimization, the path length is taken as the objective function. By adjusting the path parameters (such as the small arc radius R and the large arc radius 2R), the turnaround path is optimized to minimize the total path length.

Energy Function: The total length of the turnaround path L_{turn} is regarded as the energy function, and the goal is to find the parameter combination that minimizes L_{turn}.

Temperature Control: The algorithm starts with a high initial temperature, allowing the system more freedom to avoid being trapped in local minima. As the number of iterations increases, the temperature gradually decreases, allowing the system to converge to the global optimal solution.

2.4.2. Optimization Objectives and Parameter Settings

1. Optimization Objective: Minimize the total length of the "S"-shaped turnaround path. The objective function is the same as Equation (6). The goal is to find the optimal small arc radius R and large arc radius 2R that minimize the total length L_turn.

ISSN: 1813-4890

2. Initial Parameter Setup: Set the initial small arc radius R_{init} and initial temperature T_{init}. Additionally, set the minimum temperature T_{min} and temperature decay coefficient α, which control the cooling rate of the temperature.

2.4.3. Generation of Neighbor Solutions and Acceptance Criteria

In each iteration, the SA algorithm generates a new solution, i.e., a new small arc radius R_{new}. The steps for generating neighbor solutions and the acceptance criteria are as follows:

Neighbor Solution Generation: A new radius R_{new} is generated by adding a random perturbation ΔR to the current radius R_{curren}. The perturbation range gradually decreases as the temperature decreases, ensuring finer search at lower temperatures.

Objective Function Calculation: Compute the objective function values for the current and new solutions, i.e., calculate $L_{turn}(R_{current})$ and $L_{turn}(R_{new})$.

Acceptance Criteria: If the objective function value of the new solution is smaller (i.e., the path length is shorter), the new solution is accepted directly. If the new solution's objective function value is larger, the new solution is accepted with a certain probability, determined by the formula:

$$P = \exp\left(-\frac{\Delta L}{T}\right) \tag{7}$$

Where $\Delta L = L_{turn}(R_{new}) - L_{turn}(R_{current})$ and T is the current temperature. This criterion allows the algorithm to escape local minima and explore the global optimum.

2.4.4. Temperature Decay and Iteration Termination Conditions

After each iteration, the temperature is updated according to the decay coefficient α, following the equation $T_{new} = \alpha * T_{current}$. As temperature decreases, the probability of accepting worse solutions diminishes, allowing algorithm to converge to the optimal solution[9,10]. The algorithm stops when either: ①The current temperature falls below the minimum temperature T_{min}.②the maximum number of iterations is reached, ensuring sufficient exploration of the solution space.

3. Result

3.1. Path Modeling Results

The visualization of the dragon dance team's spiral inward motion with a 0.55-meter constant pitch in a clockwise direction is shown (Fig. 1). Additionally, a dynamic display of the movement at each second can be provided. This visualization helps better understand the inward spiral motion of the dragon dance team and the trajectory of the dragon head.

3.2. Collision Detection Model Results

The dragon dance team's trajectory follows a spiral path. By calculating whether the boards of the inner and outer layers of the spiral overlap, we can determine if a collision has occurred (Fig. 2). The dragon team spirals inward until one of the four corner points of an adjacent bench overlaps with another bench's corner point, indicating a collision at that moment.

ISSN: 1813-4890

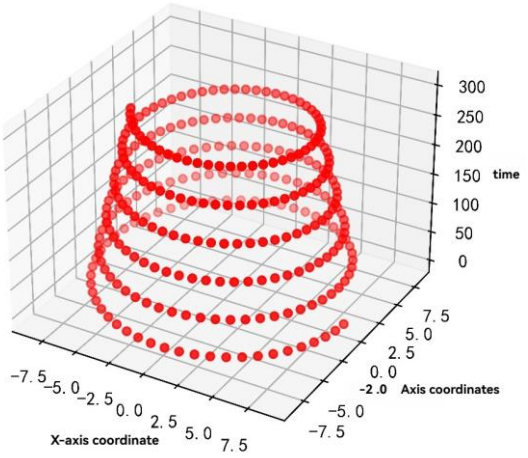

Fig. 1 Changes in the position of the dragon head over time

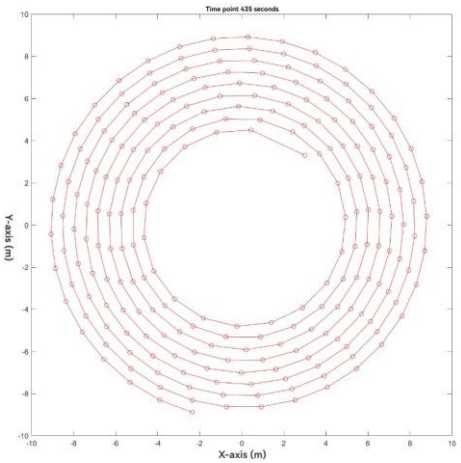

Fig. 2 Critical state at the moment of collision

3.3. Turnaround Space Model Verification and Simulation

Using Matlab, the inward motion of the dragon dance team with a 0.55-meter constant pitch is visualized. The pitch size is iteratively adjusted to determine when the dragon head reaches the turnaround space, enhancing understanding of the motion and trajectory.

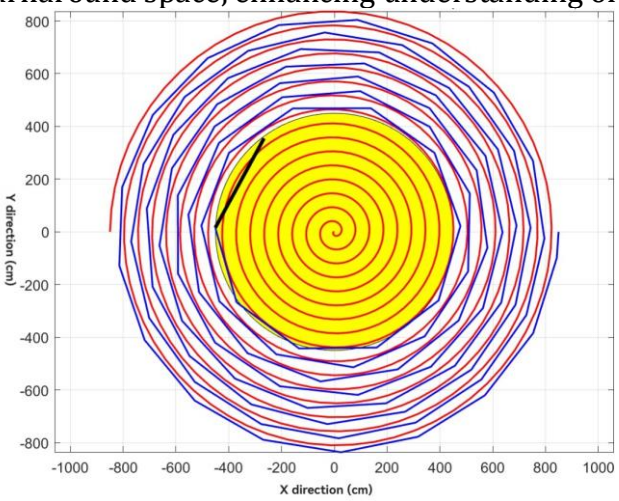

Fig.3 Constant pitch curve: entering the turnaround space

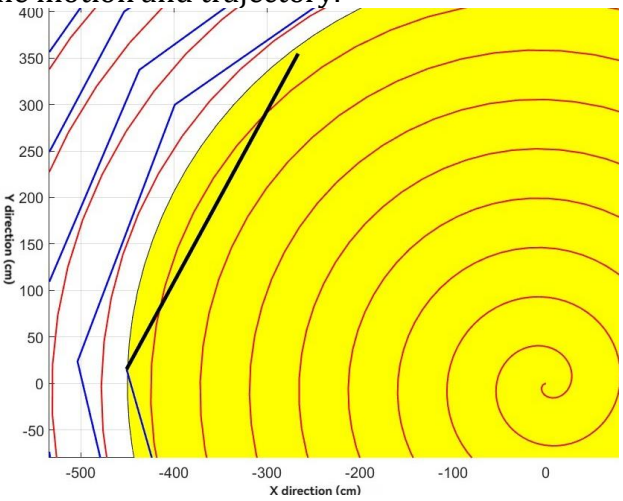

Fig.4 Constant pitch curve: magnified view of entering the turnaround space

3.4. Simulated Annealing Optimization Results and Analysis

The simulated annealing algorithm is employed to optimize and calculate the minimum total length of the turnaround curve, yielding a shortest path of 13.7212 meters. The large arc (blue portion) represents the beginning of the S-shaped turnaround path, indicating the moment the dragon head is about to change direction. The small arc (red portion) marks the end of the path, as the dragon head prepares to exit the spiral.

ISSN: 1813-4890

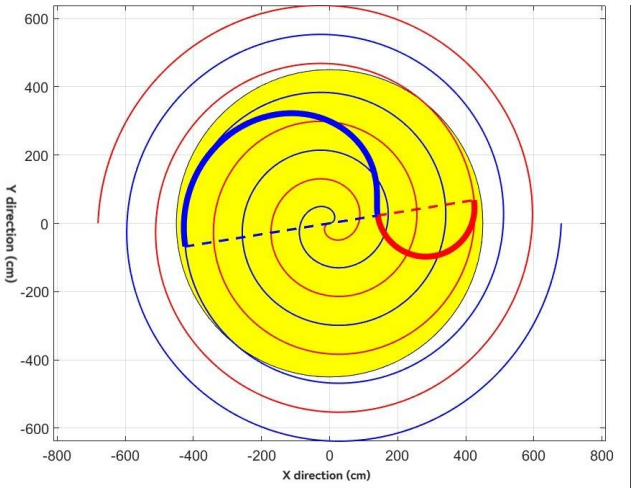

Fig.5 Length of the minimum turnaround curve

4. Summary

This study establishes a mathematical model and optimization algorithm to precisely model and optimize the path of the "bench dragon" system. The minimum pitch is determined to be 0.4597 meters, and the turnaround path is optimized to 13.7212 meters. The optimization minimizes path length while ensuring system stability. Experiments verify the algorithm's effectiveness, providing a scientific basis for path optimization and spatial planning in similarly complex dynamic systems.

Acknowledgements

The authors gratefully acknowledge the financial support from the Undergraduate Teaching Reform Research Project of Hunan Provincial Education Commission (202401001521).

References

[1] Sarimveis H, Patrinos P, Tarantilis C D, Kiranoudis C T. Dynamic modeling and control of supply chain systems: A review. Computers & Operations Research, 2008, 35(11): 3530-3561.

[2] Lee C W, Lee J H, Cha B J, Kim H Y, Lee J H. Physical modeling for underwater flexible systems dynamic simulation. Ocean Engineering, 2005, 32(3-4): 331-347.

[3] Oussar Y, Dreyfus G. How to be a gray box: dynamic semi-physical modeling. Neural Networks, 2001, 14(9): 1161-1172.

[4] Cannon R H. Dynamics of physical systems. Courier Corporation, 2003.

[5] Bertsimas D, Tsitsiklis J. Simulated annealing. Statistical Science, 1993, 8(1): 10-15.

[6] Yao X. A new simulated annealing algorithm. International Journal of Computer Mathematics, 1995, 56(3-4): 161-168.

[7] Park M W, Kim Y D. A systematic procedure for setting parameters in simulated annealing algorithms. Computers & Operations Research, 1998, 25(3): 207-217.

[8] Zomaya A Y, Kazman R. Simulated annealing techniques. In: Algorithms and theory of computation handbook: general concepts and techniques, 2010: 33-33.

[9] Delahaye D, Chaimatanan S, Mongeau M. Simulated annealing: From basics to applications. Handbook of Metaheuristics, 2019: 1-35.

[10] Azizi N, Zolfaghari S. Adaptive temperature control for simulated annealing: a comparative study. Computers & Operations Research, 2004, 31(14): 2439-2451.