Research on Bench Dragon Path Planning Based on Geometric Kinematics Model

Bohan Zhang[1,a,*], Yuxiao Shang[2,b], Xiaolu Lin[3,c]

[1]Institute of Information Science and Engineering, Fudan University, Handan Road, Shanghai, China

[2]Institute of Computer Science, Fudan University, Handan Road, Shanghai, China

[3]Institute of Microelectronics, Fudan University, Handan Road, Shanghai, China

[a]bohanzhang1022@gmail.com, [b]23307130456@m.fudan.edu.cn, [c]23307130455@m.fudan.edu.cn

*Corresponding author

Keywords: Bench Dragon, Spiral Line, Path Planning, Collision Detection, Speed Optimization, Computer Simulation

Abstract: This paper focuses on the modeling, solution, and optimization of the motion of a dragon dance team along an equidistant spiral trajectory. First, a kinematic model based on the spiral equation is constructed, and formulas for the displacement, velocity, and acceleration of the dragon head and body are derived. Through numerical simulations, the motion trajectories and velocities of the dragon head, body, and tail at different time nodes are calculated, with the results presented in the form of tables and graphs. On this basis, a collision detection mechanism is proposed, using geometric relationships to determine if the dragon's benches overlap. The collision between the dragon head and the following segments of the dragon body is detected using the triangular area method. Numerical simulation results show that the first collision occurs at 412.478 seconds, making it impossible for the dragon to continue along the spiral path. Subsequently, research is conducted to optimize the model, focusing on the pitch parameter and the turnaround curve path. Using incremental search and the bisection method, the minimum pitch is determined to be 45.035 cm, enabling the dragon team to successfully follow the trajectory without collisions. The optimization of the turnaround path reveals that when the tangent distance between the spiral and arc is 427.5 cm, the dragon team achieves the optimal path. Additionally, considering the peak velocity of the dragon body, the dragon head's velocity and path parameters are optimized to limit the maximum velocity of the dragon body to within 2 m/s, ensuring smooth and controllable movement. The results of this study provide theoretical support and algorithmic tools for path planning, formation control, and collision detection in dragon dance performances. The constructed model and optimization methods can be applied to motion control and path optimization for other multi-segment rigid bodies.

1. Introduction

The dragon dance is a traditional folk performance art with rich cultural connotations and strong visual impact. The path design and formation control of the performance are crucial to its overall effect. During the dragon dance, high coordination is required between the head and the body, especially when navigating complex paths such as spirals. Ensuring the smooth movement of each bench segment and avoiding collisions are key issues that need to be addressed.

This study focuses on the movement characteristics of the dragon dance team along an equidistant spiral trajectory. A kinematic model based on the spiral equation is proposed, and methods for calculating the displacement and velocity of the dragon head and body are established. By constructing geometric relationships and a collision detection mechanism, collision risks between the benches are predicted and detected. Furthermore, pitch optimization, turnaround path optimization, and velocity optimization strategies are proposed. Using numerical simulation methods, the optimal pitch and turnaround curve are determined to ensure that the maximum peak velocity of the dragon body is controlled within a reasonable range. The results of this study provide theoretical support and

  DOI: 10.25236/icmmct.2025.021

technical assurance for path planning and motion control in dragon dance performances.

2. Construction of Spiral Motion Model Dynamic Model

2.1. Construction of the Kinematic Model

Spiral Equation:

$$r=\frac{p}{2\pi}\theta \tag{1}$$

Arch Length Formula for a Spiral:

$$ds=\sqrt{(dr)^2+(rd\theta)^2} \tag{2}$$

$$\frac{v_r}{\omega}=\frac{\frac{dr}{dt}}{\frac{d\theta}{dt}} \tag{3}$$

Then, we will get:

$$v_\tau=\sqrt{{v_r}^2+r^2\omega^2}=\sqrt{1+(\frac{2\pi}{p}r)^2}\cdot v_r=\sqrt{1+(\frac{2\pi}{p}r)^2}\cdot\frac{dr}{dt} \tag{4}$$

$$t=\int_{r_A}^{r}\sqrt{(\frac{p}{2\pi})^2+r^2}\ dr \tag{5}$$

Take the derivative of both sides with respect to t to obtain the tangential velocity v_τ.

Given that the dragon head's velocity v_τ=1m/s and the pitch p = 0.55m, integrating to obtain the function of t with respect to r.

$$t=f(r)=g(\theta) \tag{6}$$

Then, calculate the various data for the dragon head per second: r_0, θ_0, x_0, y_0.

Next, calculate the position r_i, θ_i and velocity v_{r_i} of the dragon body:

$$v_{\tau_i}=\sqrt{1+(\frac{2\pi}{p}r_i)^2}\frac{dr_i}{dt} \tag{7}$$

Calculate the following functions every second:

$$\begin{cases} r_1=p\theta_1/2\pi & ① \\ {l_0}^2={r_0}^2+{r_1}^2-2r_0r_1\cos(\theta_1-\theta_0) & ② \end{cases} \tag{8}$$

There are multiple solutions, but the unique solution can be determined using the given conditions $\theta_0<\theta_1<\theta_0+\pi$.

Taking the derivative of both sides of equation 2 with respect to t gives $\frac{dr_1}{dt}$. Next, iterate on r, θ, $\frac{dr}{dt}$.

$$\begin{cases} r_{i+1} = p\theta_{i+1} / 2\pi \\ l_i^2 = r_i^2 + r_{i+1}{}^2 - 2r_i r_{i+1}\cos(\theta_{i+1} - \theta_i) \\ \dfrac{dr_{i+1}}{dt} = \dfrac{2r_i \dfrac{dr_i}{dt} - 2r_{i+1}\dfrac{dr_i}{dt}\cos(\dfrac{2\pi}{p}r_{i+1} - \theta_i) - 2r_i r_{i+1}\sin(\dfrac{2\pi}{p}r_{i+1} - \theta_i)\dfrac{d\theta_i}{dt}}{-2r_{i+1} + 2r_i\cos(\dfrac{2\pi}{p}r_{i+1} - \theta_i) - 2r_i r_{i+1}\sin(\dfrac{2\pi}{p}r_{i+1} - \theta_i)\dfrac{2\pi}{p}} \end{cases} \tag{9}$$

$(\theta_i < \theta_{i+1} < \theta_i + \pi)$

The data for each segment of the dragon body is obtained per second.

2.2. Model Solution

2.2.1. Basic Parameter Setup

The code initializes the workspace by clearing it and sets various parameters for calculating the trajectory of the dragon dance team. Key parameters include the hole spacing, which refers to the distance between the holes in the dragon's body and the dragon head, and is linked to the geometric size of the benches. The total number of benches is 223, with 1 for the dragon's head, 221 for the body, and 1 for the tail. The dragon's head is set to move at a speed of 100 cm/s (1 m/s), and the pitch, which determines the height of the spiral, is set at 55 cm. The dragon's movement follows a spiral path with 22 evenly spaced turns [1].

2.2.2. Spiral Trajectory Calculation

The code generates the trajectory of the dragon dance team by calculating the position of each point using the formula for an evenly spaced spiral. First, the polar coordinates (angle and radius) for each point are determined using the spiral formula. These coordinates are then converted into Cartesian coordinates, which give the precise position of the spiral at each time point. The path is divided into segments, and the distance between adjacent points is calculated by finding the vector difference. The total travel distance for each segment of the dragon is tracked through cumulative calculations, providing a detailed representation of the dragon's movement.

2.2.3. Dragon Team's Position and Velocity Calculation

To calculate the position and velocity of the dragon dance team at different time points, the code utilizes an iterative process to update the bench positions. Initially, a list called Loong_list is created to store the data for each bench, including its length and its X and Y coordinates. This list is updated at each time step to reflect the current position of each bench in the sequence [2].

Starting from the dragon head, the position of each bench is updated along the spiral path based on the point number within the spiral. As the dragon moves, the latest position for the head, body, and tail is continuously updated. The position of each hole is recalculated by approximating the vector from one point to the next as a tangent vector. This tangent vector is scaled by the length of the bench, and the coordinates for the next hole are initialized accordingly. This iterative process allows the code to track the dragon's movement through the spiral with high precision, updating its trajectory in real-time.

The code iteratively calculates the distance from each hole to the spiral and finds the two closest points on the spiral. The hole's position is then updated to be closer to the line segment, with the coordinates scaled by the hole distance.

Bench Velocity Calculation: The velocity at the target time is computed using the average velocity of the previous two and next two positions, with a uniform acceleration model. The time scale is proportional to the velocity difference at the previous and next times.

2.2.4. Result Output

The code is designed to capture data at specific time points (e.g., 0s, 60s, 120s) and for specific benches (e.g., the 1st, 51st, 101st, 151st, and 201st of the dragon body, and the tail). These data are saved in location_results and velocity_results as the output.

The model is run with the geometric parameters for the dragon benches (length, width, hole diameter, etc.), the dragon head speed, and the spiral pitch. The MATLAB code is used for numerical simulation, and graphs are generated. This solution models the motion based on a spiral and addresses the calculation of position and velocity along the spiral path for the dragon dance team.

The extracted data is as shown in Table 1 and Table 2.

The visualized data is as shown in Figure 1, Figure 2 and Figure 3.

Table 1 The Result of Velocity

	0 s	60 s	120 s	180 s	240 s	300 s
Dragon Head	1	1	1	1	1	1
The 1st Dragon Body	0.999971	0.999961	0.999945	0.999916	0.99986	0.999715
The 51st Dragon Body	0.999757	0.999584	0.999577	0.999278	0.999017	0.998026
The 101st Dragon Body	0.999594	0.999406	0.999298	0.998689	0.998457	0.997284
The 151st Dragon Body	0.99947	0.999056	0.998936	0.998528	0.998149	0.996841
The 201st Dragon Body	0.99935	0.998895	0.99885	0.998387	0.99788	0.996577
Dragon Tail Back	0.999336	0.998869	0.998808	0.998322	0.997777	0.996441

Table 2 The Result of Position

	0 s	60 s	120 s	180 s	240 s	300 s
Dragon Head x (m)	8.8	5.799174	-4.084885	-2.963577	2.594446	4.420293
Dragon Tail y (m)	0	-5.771073	-6.304436	6.094791	-5.356743	2.320377
The 1st Dragon Body x (m)	8.363819	7.456715	-1.445485	-5.237091	4.821178	2.459534
The 1st Dragon Body y (m)	2.826543	-3.44039	-7.405838	4.359647	-3.56197	4.402444
The 101st Dragon Body x (m)	2.914043	5.687137	5.361953	1.898825	-4.917348	-6.237718
The 101st Dragon Body y (m)	-9.918288	-8.001326	-7.557587	-8.471601	-6.379872	3.936001
The 151st Dragon Body x (m)	10.861706	6.682227	2.388682	1.005091	2.965328	7.040715
The 151st Dragon Body y (m)	1.828847	8.134571	9.7274	9.424752	8.39972	4.393039
The 201st Dragon Body x (m)	4.554985	-6.619733	-10.62719	-9.287676	-7.457093	-7.458617
The 201st Dragon Body y (m)	10.725163	9.025479	1.359744	-4.246754	-6.180775	-5.263435
Dragon Tail Back x (m)	-5.305318	7.364636	10.97431	7.383815	3.240964	1.784952
Dragon Tail Back y (m)	-10.67664	-8.79789	0.843584	7.49244	9.469353	9.301172

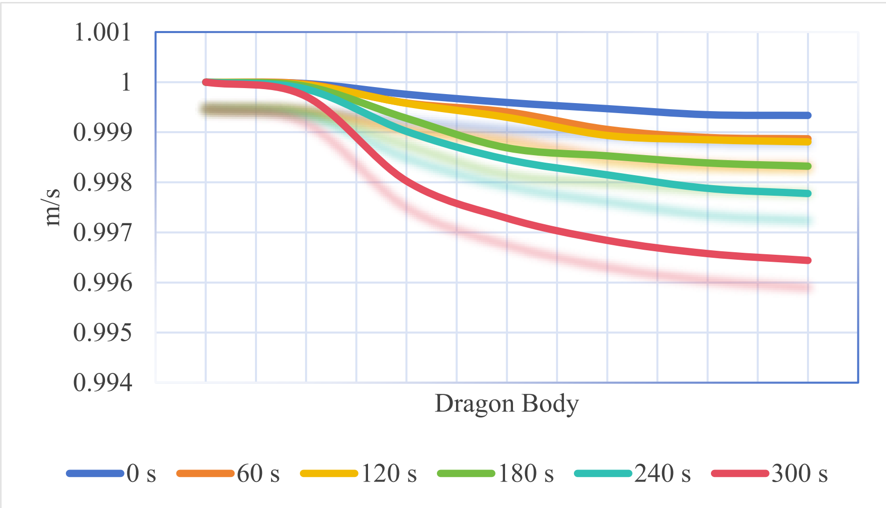

Figure 1 Relationship between the Velocity of Some Dragon Body Sections and Time

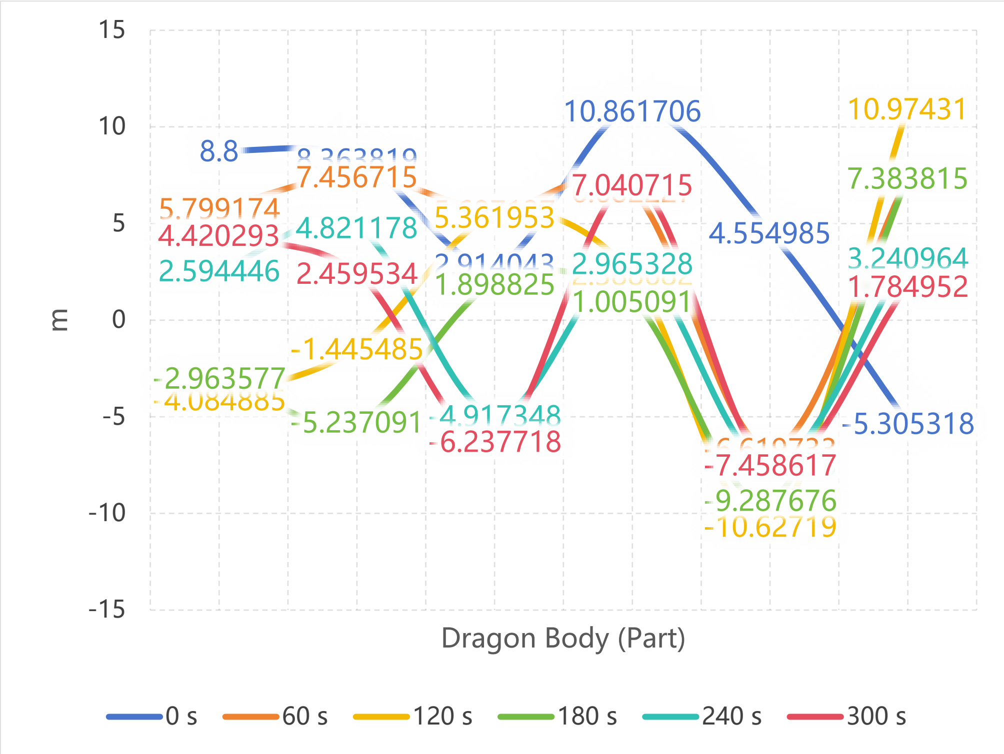

Figure 2 Relationship between the Horizontal Coordinates of Some Dragon Body Sections and Time

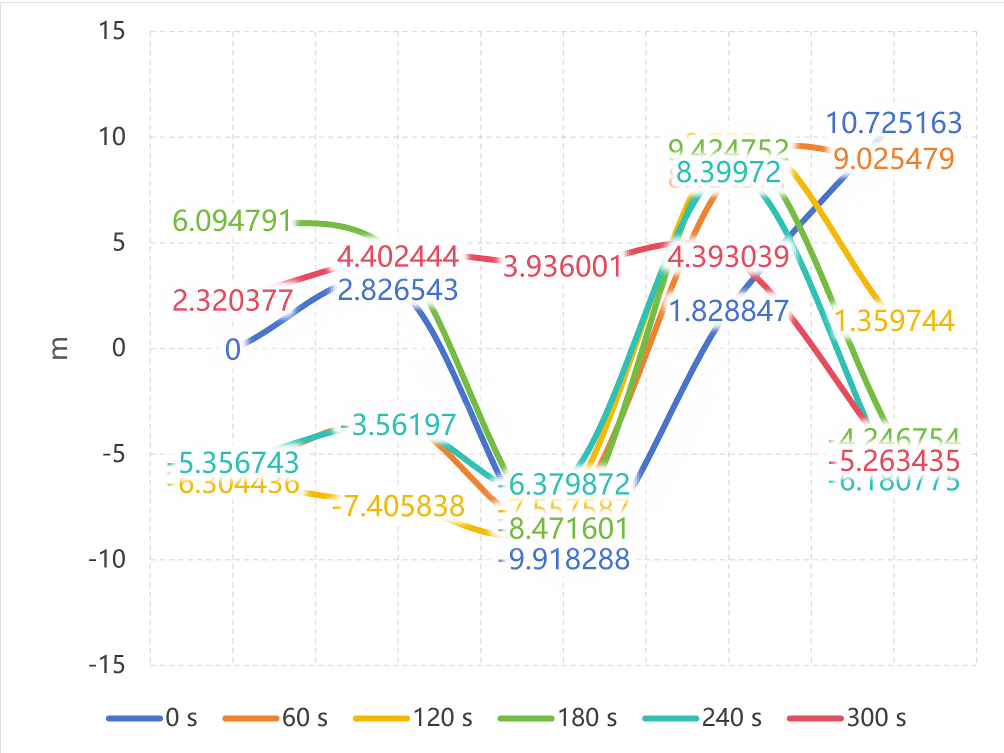

Figure 3 Relationship Between the Longitudinal Coordinates of Some Dragon Body Sections and Time

3. Collision Detection Mechanism

3.1. Geometric Relationship and Collision Judgement

The model can simulate the trajectory of the dragon benches at each time point and generate graphs. During the simulation, the code detects whether a collision occurs. The principle is based on a point and four triangles formed by the edges of a rectangle. The areas of these triangles are compared with the area of the rectangle. If the sum of the triangle areas is less than or equal to the area of the rectangle (within a certain threshold range), the system will determine that a collision has occurred and terminate the calculation, thus identifying the time at which the dragon can no longer continue to coil [3].

Since the dragon head has the largest area, its movement plane will inevitably cover all subsequent plates' movement planes. A collision will always occur between the dragon head and the other benches. The first collision detection hole (hole1) is set between two holes of the dragon head, and the second detection hole (hole2) is traversed from hole1 + 3 until the dragon tail.

If the collision condition is met: Further calculation can determine if the two benches have collided:By determining the unit direction vector and the coordinates of the rectangle's vertices, the four vertices are traversed. The triangle area method is used to judge whether there is overlap [4].

3.2. Simulation Experiment and Result Verification

Through numerical simulation, at 412.478 seconds, the distance between the benches becomes smaller than the set threshold, and the motion stops. The position information at this point is saved in location_results_end, and the velocity information is saved in velocity_results_end.

Through collision detection, we found that the dragon benches could not avoid a collision as the motion continued, as shown in Figure 4 and Figure 5, so the termination time is 412.478 seconds.

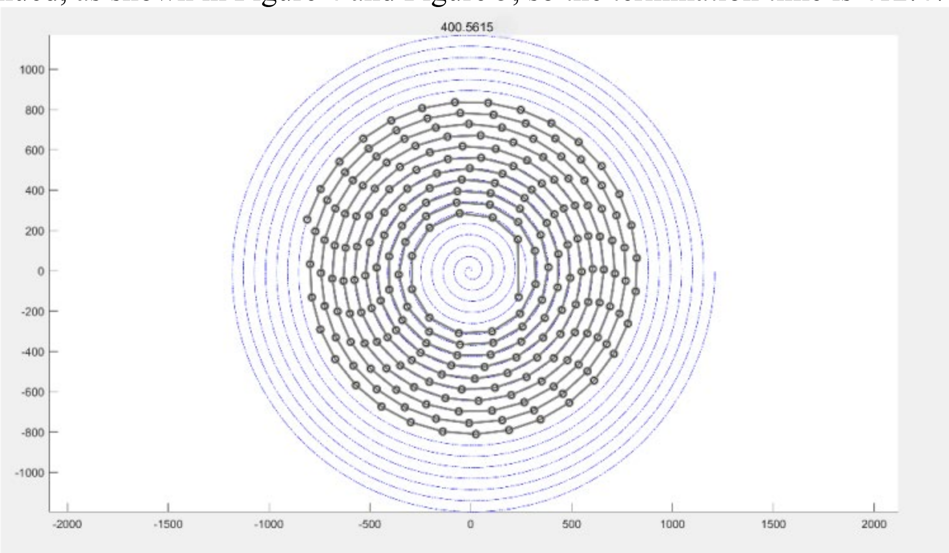

Figure 4 Simulation Image When No Collision Occurs

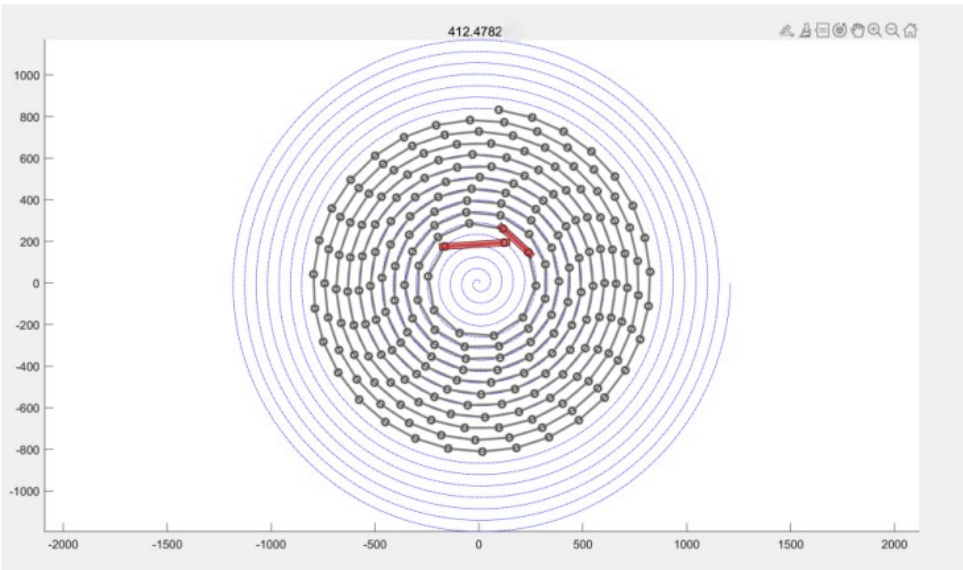

Figure 5 Simulation Image When Collision Occurs

4. Model Optimization

4.1. Pitch Parameter Optimization

The goal is to minimize the likelihood of collision in the simulation model by adjusting the pitch. The determination of the dragon's motion trajectory and collision detection is the same as previous discussion. On this basis, the collision detection termination time is limited, and the pitch is adjusted step by step.

To find the smallest pitch that avoids collision, the pitch value must be continuously adjusted, and the simulation should be rerun each time. For each run, the trajectory and motion of the dragon are calculated based on the given pitch. Collision detection is performed before the dragon head reaches the boundary for turning. If the collision detection module outputs a collision, it indicates that the pitch is too small, and it can be increased. If no collision occurs, the pitch can be further reduced until the smallest non-colliding value is found.

We simulate that a collision occurs when the pitch is 45cm. Thus, a step-by-step incremental search algorithm is used, starting with an initial pitch of 45cm and a step size of 0.02cm, until no collision is detected. The pitch is found to be 45.04cm. Then, the binary search method is applied to gradually refine the smallest pitch that avoids collision.

The final result is a pitch of 45.035cm, which allows the dragon head to move along the spiral trajectory up to the boundary of the turning space without causing a collision.

4.2. Optimization of the Turnaround Path

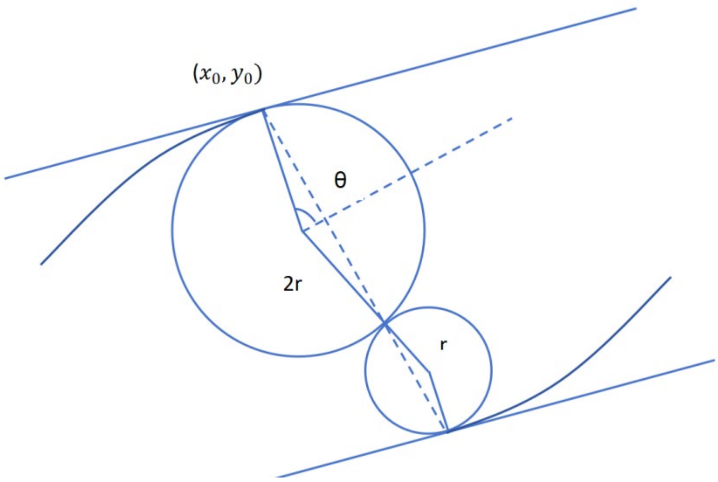

Figure 6 Simple Diagram of the Turnaround Path

As shown in Figure 6, the spiral Equation in the Cartesian Coordinate System [5].

$$\sqrt{x^2+y^2}=\frac{p}{2\pi}\arctan\frac{y}{x} \tag{10}$$

Taking the derivative of both sides with respect to x, we get:

$$k_{l_1}=k_{l_2}=\frac{dy}{dx}\Big|_{(x_0,y_0)}\triangleq k \tag{11}$$

The equations of straight lines are:

$$\begin{aligned} l_1 &: kx-y+y_0-kx_0=0 \\ l_2 &: kx-y-y_0+kx_0=0 \\ l_3 &: x+ky-x_0-ky_0=0 \\ l_4 &: x+ky+x_0+ky_0=0 \end{aligned} \tag{12}$$

$$d_{12} = \frac{2|y_0 - kx_0|}{\sqrt{k^2+1}} = 3R + 2R\cos(\pi - \theta)$$

$$d_{34} = \frac{2|x_0 + ky_0|}{\sqrt{k^2+1}} = 2R\sin(\pi - \theta) \tag{13}$$

The arc length of the U-turn curve is:

$$S = 3R\theta \tag{14}$$

While,

$$\begin{cases} \mathrm{k} = \dfrac{\dfrac{px_0}{2\pi\sqrt{x_0^{\,2} + y_0^{\,2}}} - y_0}{\dfrac{py_0}{2\pi\sqrt{x_0^{\,2} + y_0^{\,2}}} + x_0} \\ \theta = 2\arctan\left|\dfrac{y_0 - kx_0}{x_0 + ky_0}\right| \end{cases} \tag{15}$$

The simulation begins by initializing the parameters, including the geometric information of the spiral line and circular arc. The complete trajectories of both the spiral and circular arc are then generated, and the position of each node is calculated. Using an approximation algorithm, the exact position of each bench is determined. The velocity is calculated through position differences, and the position and velocity information at each time point are stored. During the simulation, the dragon dance trajectory is plotted in real-time while checking for potential collisions. If a collision occurs, the simulation stops, and the position of the collision is output; if no collision is detected, the simulation continues until the specified time range is completed. For optimization, the circular arc can be adjusted by modifying the p_0 variable, which allows for further fine-tuning of the dragon's head turn and optimizing the entire dragon dance team's route. Collision handling can also be enhanced, such as by recording detailed information when a collision occurs or adjusting the trajectory to avoid future collisions. The final optimal route is chosen based on the scenario where the turning starting point is 427.5 cm from the origin, resulting in the shortest turning curve.

4.3. Optimization of Travelling Speed

The optimization of the dragon dance model focuses on adjusting the dragon head's velocity to ensure body speed stays below 200 cm/s. Initially, the body velocity often exceeded the head's due to the path's geometry. The entry and exit of the spiral are symmetric, and the spiral and circular arc are tangent at 427.5 cm from the center, leading to velocity increases during transitions. The radii ratio of 2:1 at the spiral's entry and exit tightens the curve, further increasing body speed. With the dragon head moving at 1 m/s, these geometric factors caused peak body velocities of up to 5.8 m/s, particularly at sharper curves.

To minimize the body's peak velocity, adjustments were made. The tangent point was shifted to 450 cm from the center, and the radii ratio was reduced to 1:1, smoothing the path and reducing sharp transitions. This achieved a consistent velocity profile for the dragon, with the head's speed remaining at 1 m/s. As a result, the peak body velocity dropped significantly. Simulations showed that at 7.0404 seconds, the maximum body velocity was reduced to 1.082 m/s. By increasing the dragon head's speed to 1.848 m/s, the body's maximum velocity reached 1.9998 m/s at 3.8353 seconds, ensuring it remained under the 2 m/s limit. These optimizations ensured smooth and controlled movement for the dragon dance.

5. Conclusion

This paper presents a comprehensive study on the modeling, solution, and optimization of the motion of a dragon dance team following an equidistant spiral trajectory. A kinematic model based on the spiral equation is developed, with displacement, velocity, and acceleration equations derived for the dragon head and body. Numerical simulations are conducted to calculate the motion trajectories and velocities of the dragon at different time points, and the results are displayed through tables and graphs. A collision detection mechanism is introduced to identify overlaps between the dragon's benches, and a triangular area method is applied to detect collisions between the dragon head and its following segments. The first collision occurs at 412.478 seconds, preventing further movement along the spiral path.

Optimization research follows, focusing on the pitch parameter and the turnaround curve path. The minimum pitch of 45.035 cm is found through the incremental search and bisection method, allowing the dragon to navigate the spiral path without collisions. The optimal turnaround path is achieved when the tangent distance between the spiral and arc is 427.5 cm. Further, the maximum velocity of the dragon body is limited to 2 m/s through optimization of the dragon head's velocity and path parameters, ensuring smooth and controllable movement.

The findings of this study offer valuable theoretical insights and algorithmic tools for path planning, formation control, and collision detection in dragon dance performances. The proposed model and optimization techniques can be adapted for the motion control and path optimization of other multi-segment rigid bodies, expanding their potential applications beyond dragon dance.

References

[1] Heermann D W, Heermann D W. Computer-simulation methods[M]. Springer Berlin Heidelberg, 1990.

[2] Wilkins M L. Computer simulation of dynamic phenomena[M]. Springer Science & Business Media, 2013.

[3] Lin M, Gottschalk S. Collision detection between geometric models: A survey[C]//Proc. of IMA conference on mathematics of surfaces. 1998, 1: 602-608.

[4] Ericson C. Real-time collision detection[M]. Crc Press, 2004.

[5] Nakamura T, Sekine T, Yokokawa S. Analytical method of spiral line[J]. Electronics and Communications in Japan (Part I: Communications), 1983, 66(10): 49-57.