刘一帆，19357599217，24 级医学信息与工程学院，大数据一班

2024A 题 板凳龙

问题假设:

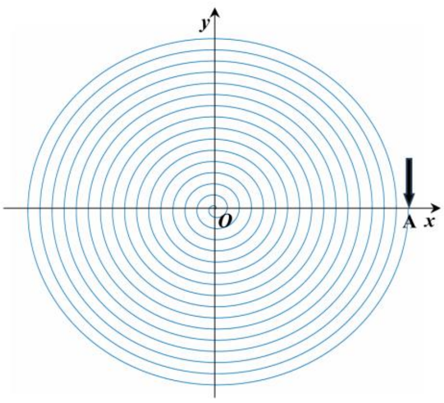
1.假设两板凳刚刚接触但不发生碰撞，在此我们定义它为不发生碰撞。

2.假设板凳厚度忽略不计。

3.假设舞龙期间不受人体碰撞影响。

一．准备工作

通过读解题目，可知龙头，龙身，龙尾的把手中心均在等距螺旋线上，并且舞龙期间进行标准的阿基米德螺线运动。因为极坐标中的极角和极径常用于旋转物体运动表示，所以我们可以用极坐标系中表示阿基米德螺线方程进行龙头的位移和速度的求解。



以螺旋线的中心为极坐标的极点，根据某一时刻该点的转动角度和把手与极点的距离，我们可以把M_i表示为(r_i,θ_i),其中r_i表示从龙头开始后的极径，θ_i表示对应的极角，$i=1,2,3\dots\dots 224$。以下为极坐标与直角坐标的关系式。

$$X = rcos\theta$$

$$Y = rsin\theta$$

$$r = \frac{a}{2\pi}\theta \ , \theta\epsilon[0,32\pi]$$

根据题意可知螺距$a = 0.55\ (m)$，其中r为螺线上某点的极径,θ为某点的极角，根据这个式子可知通过极角可以确定极径，因此极角的确定影响着我们整个模型的建立。对各把手位置和速度的求解核心就是找到把手θ与时间t的关系。对于第i个把手$M_i(r_i,\theta_i)$,可由下列公式，得到直角坐标系上的位置。

$$\begin{cases} x_i = r_i cos\theta_i \\ y_i = r_i sin\theta_i \end{cases}$$

为方便书写，我将在接下来的模型中以极坐标的形式表明把手坐标，在计算左后结果时，使用上述公式得出二维直角坐标系的坐标。

问题一 龙头运动模型

在舞龙期间，各板凳手把处受力的作用，让龙头，龙身，龙尾紧密相连，并且龙身板长长度一样，这将说明研究龙头的运动状态和位置，与后续板凳的分析大体一致。据上文，可知该模型的核心是在$\theta_1(0)$这一初始条件下，确定任意时间极角$\theta_1(t)$随时间的函数关系:

在舞龙期间，龙头前把手始终以$v_0 = 1m/s$的速度进行移动，则在dt时间其走过的弧长可表示为:

$$v_0 dt = ds$$

以下是相关证明:

$$ds = \sqrt{(dx)^2 + (dy)^2}$$

$$v_0 dt = \sqrt{(dx)^2 + (dy)^2}$$

$$dx = (\frac{a}{2\pi}cos\theta\ - \frac{a}{2\pi}\theta sin\theta)d\theta$$

$$dy = (\frac{a}{2\pi}sin\theta + \frac{a}{2\pi}\theta cos\theta)d\theta$$

$$v_0 dt = \frac{a}{2\pi}\sqrt{1+\theta^2}d\theta$$
$$\frac{d\theta}{dt} = \frac{2\pi}{a}v_0\frac{1}{\sqrt{1+\theta^2}}$$

通过上述式子我们得到了极角与时间的关系，再结合我们龙头的初始值，我们可以写出龙头的极角模型:

$$\begin{cases} \dfrac{d\theta}{dt} = \dfrac{2\pi}{a}v_0\dfrac{1}{\sqrt{1+\theta^2}} \\ \theta_1(0) = 32\pi \end{cases}$$

通过上述公式我们继续推导:

$$\sqrt{1+\theta^2}d\theta = -\frac{2\pi}{a}v_0 dt \quad (\text{分离变量})$$
$$\int\sqrt{1+\theta^2}d\theta = -\frac{2\pi}{a}v_0 \qquad (\text{两边积分})$$
$$\frac{1}{2}\left(\theta\sqrt{1+\theta^2}+\ln\left(\theta+\sqrt{1+\theta^2}\right)\right)+\frac{2\pi}{a}v_0 t-\frac{1}{2}\left(\theta\sqrt{1+\theta^2}+\ln\left(\theta+\sqrt{1+\theta^2}\right)\right) = 0(\text{隐式方程})$$

由于上述式子十分复杂，不能写出关于$\theta_1(t)$的显性形式，因此我们把龙头把手的极坐标表示成隐性形式；稍后我们以二分法为解决方法对此进行求解。

通过公式()我们可以推到出龙头位置的相关关系:

$$\begin{cases} x = \dfrac{a}{2\pi}\theta_0 cos\theta_0 \\ y = \dfrac{a}{2\pi}\theta_0 sin\theta_0 \end{cases}$$

通过龙头前把手位置递推之后把手的位置

龙身和龙尾紧密相连，后把手受前把手的状态影响，由此我们将采用递推的方法，以龙头的位置为起始参数，每个板凳前后各把手间距为距离限制，以把手为圆心画圆，与螺线相交，通过不断的递归更新交点数据，直至推出龙身与龙尾上所有把手的坐标值。

设以把手$M(X_i, Y_i)$为圆心，以板凳前后两孔间的距离 R 为半径做一个圆，又因为所有把手位置坐标都满足螺旋线的方程，所以经过以公式可得到圆在极坐

标下的方程式:

$$\begin{cases}(x-x_i)^2+(y-y_i)^2=R^2\\ x=\dfrac{a}{2\pi}\theta cos\theta\\ y=\dfrac{a}{2\pi}\theta sin\theta\end{cases}$$

$$\xRightarrow{\text{推导出}}\left(\frac{a^2}{4\pi^2}\left({\theta_{i+1}}^2+{\theta_i}^2-2\theta_i\theta_{i+1}\cos(\theta_{i+1}-\theta_i)\right)=R^2\right)$$

其中，θ_i为第i个把手M的极角，()方程式描述了螺线线下一个把手的极角与把手M之间的关系。

由于龙头出的半径与龙身龙尾的不同，我们分开讨论。

1. 龙头与龙身交界处的位置关系

龙头前把手的极坐标为$M_1(r_{1,}\theta_1)$，龙头板凳两口间的长度与龙身龙尾不同，所以，我们对方程式()里的半径R更改为R_1，然后龙头后把手为$M_2(r_{2,}\theta_2)$

与龙头前把手的关系如下:

$$\begin{cases}\dfrac{a^2}{4\pi^2}\left({\theta_2}^2+{\theta_1}^2-2\theta_1\theta_2\cos(\theta_2-\theta_1)\right)=R^2\\ R_1=2.86\ (m)\end{cases}$$

2. 龙身与龙尾上所有把手处位置关系

从龙身第一节板凳开始，之后的两口间距相等，因此可以使用()方程式来描述相邻把手位置之间的递推关系。我们定义龙身与龙尾前后端孔径间距为R_2，对于把手$M_i(r_{i,}\theta_i)$与把手$M_{i+1}(r_{i+1,}\theta_{i+1})$位置关系如下:

$$\begin{cases}\dfrac{a^2}{4\pi^2}\left({\theta_{i+1}}^2+{\theta_i}^2-2\theta_i\theta_{i+1}\cos(\theta_{i+1}-\theta_i)\right)={R_2}^2\\ R_2=1.65\ (m)\\ i=2,3,\dots,224\end{cases}$$

因为龙头前把手位置随时间的关系是隐式表达式，所以递推后的所有把手的位置也是隐式关系，可以通过后续递推求解。

接下来我们将计算各把手速度。

以下是直角坐标:

$$\begin{cases} x_i = r_i cos\theta_i = \dfrac{a}{2\pi}\theta_i cos\theta_i \\ y_i = r_i sin\theta_i = \dfrac{a}{2\pi}\theta_i sin\theta_i \end{cases}$$

然后我们对x_i和y_i关于时间t求导:

$$v_{i,x} = \frac{a}{2\pi}(cos\theta_i - \theta_i sin\theta_i)\frac{d\theta_i}{dt}$$
$$v_{i,y} = \frac{a}{2\pi}(sin\theta_i + \theta_i cos\theta_i)\frac{d\theta_i}{dt}$$

为得到$\frac{d\theta_i}{dt}$，我们采用数值差分法:

$$\frac{d\theta_i}{dt} \approx \frac{\theta_i(t+\nabla t) - \theta_i(t-\nabla t)}{2\nabla t}$$

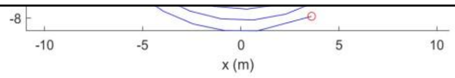
接下来通过两垂直的分速度求解v_i的大小:

$$v_i = \frac{a}{2\pi}\left|\frac{d\theta_i}{dt}\right|\sqrt{(cos\theta_i - \theta_i sin\theta_i)^2 + (sin\theta_i + \theta_i cos\theta_i)^2}$$
$$\xRightarrow[\text{化简可得}]{} v_i = \frac{a}{2\pi}\left|\frac{d\theta_i}{dt}\right|\sqrt{1+{\theta_i}^2}$$

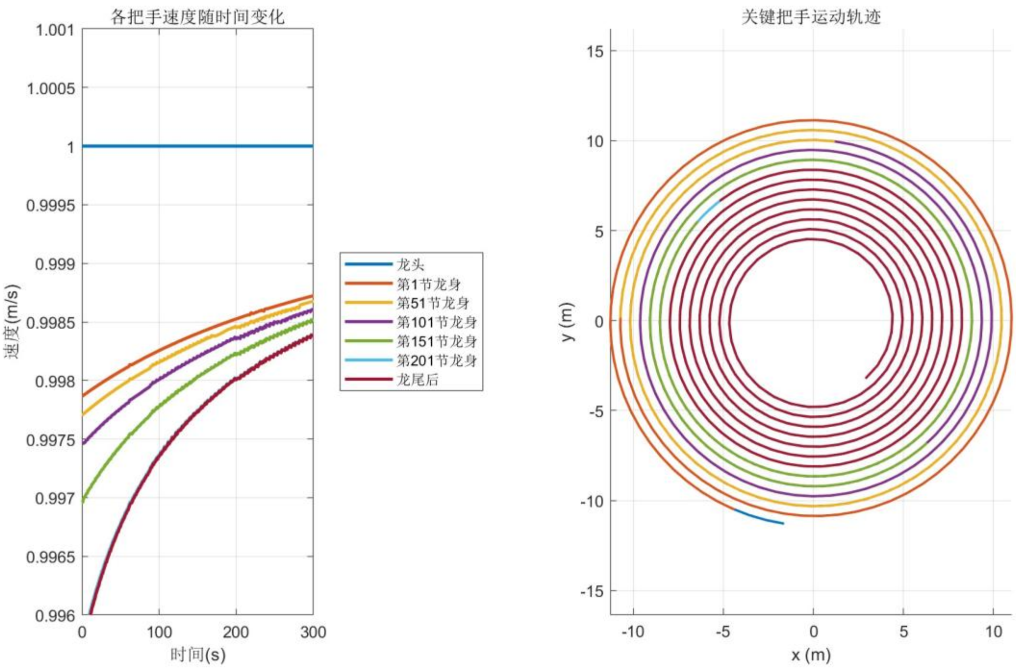
但是我们的$\theta_1(t)$并未求解，我将采用二分法进行求解，以下是我的求解过程，首先我将先找到一个区间$[b,c]$，并且龙头向着极角减小的方向运动，因此我们可设初始区间$[b,c] = [32\pi - \nabla\theta, 32\pi]$，其中$\nabla\theta$为一个小增量，验证该函数两端之积为异号。然后每次取中间值m计算该点的值，若为零，则m就是解，如果该点的值乘以b点的值小于零说明根在$[b,m]$，则令$c = m$。否则，根在$[m,c]$，$b = m$。重复以上过程，直到区间长度小于给定大小。

time	龙头	第一节龙身	第 51 节龙身	第 101 节龙身	第 151 节龙身	第 201 节龙身	龙尾后
0	1	0.99787	0.99771	0.99747	0.99696	0.99575	0.99571
60	1	0.99812	0.99799	0.99781	0.99751	0.99691	0.99689
120	1	0.99832	0.99822	0.99809	0.99788	0.99753	0.99752
180	1	0.99848	0.99840	0.99830	0.99815	0.99792	0.99791
240	1	0.99861	0.99855	0.99847	0.99835	0.99819	0.99818
300	1	0.99872	0.99868	0.99860	0.99851	0.99839	0.99838

表一 把手的相关速度



图二 板凳龙的运动轨迹



图三 把手的运动轨迹和速度变化

	0s	60s	120s	180s	240s	300s
龙头 x(m)	8.800000	8.907783	9.608291	10.33968	7.577572	-1.627202
龙头 y(m)	0.000000	2.931748	2.475535	-1.429118	-7.876271	-11.283234
第 1 节龙身 x (m)	8.306675	9.350391	9.889781	9.540303	5.261153	-4.378359
第 1 节龙身 y (m)	-2.817132	0.106195	-0.37058	-4.175132	-9.55371	-10.501767
第 51 节龙身 x (m)	-5.433677	-8.368459	-6.941356	-6.768739	-8.76549	-10.722902
第 51 节龙身 y (m)	5.739863	-1.716623	-5.941532	-6.940768	-5.259444	-10.722902
第 101 节龙身 x (m)	-5.692026	6.396952	6.559966	1.983181	-0.132706	1.188629
第 101 节龙身 y (m)	-3.949608	-4.19396	5.097729	8.694561	9.487796	9.956399
第 151 节龙身 x (m)	-5.712482	6.084415	-6.659466	-4.073061	2.995308	6.31693
第 151 节龙身 y (m)	-0.939806	2.648698	3.194107	-6.962269	-8.160971	-6.795493
第 201 节龙身 x (m)	3.912879	4.04006	-6.318473	6.480383	1.80757	-6.414029
第 201 节龙身 y (m)	-1.923295	-3.6368	-0.387861	-2.932007	7.605218	5.520782
龙尾（后）x (m)	-3.41735	0.035972	-5.4649	5.595358	-7.065025	-3.104976

龙尾（后）y (m)	0.964737	-4.81302	-1.958125	3.593624	2.196569	-7.457388

表二 把手的位置

问题二板块碰撞问题的建模与求解

问题二需要我们在问题一计算的基础上，来计算不发生碰撞（恰好发生碰撞）的时间，我们通过思考，得出了龙的首次碰撞出现在龙头。接着通过模型一的数据以及几何关系我们得出了各板凳顶点的坐标值，用于检测碰撞。基于顶点的坐标，构造线段，如果在二维坐标的线段发生重合，则说明发生碰撞，否则，未发生碰撞。

碰撞原因：因为龙头一直以速度v_0恒速向内旋转，，而板凳两把手间距不变，导致相连接的板凳间夹角不断减小，从而发生板块碰撞。

龙头首先发生碰撞，原因如下：舞龙期间，后续各节点重复前一个结点的路径，形成一个螺旋线，这证明龙头后的结点具有滞后性，龙头首先发生碰撞。

板凳的碰撞实质就是各板凳对应的矩形区域的重叠问题，因此我们要利用$M(X_i, Y_i)$的坐标来推导出该板凳对应的顶点坐标。

首先我设直线M_iM_{i+1}与水平方向的夹角为δ,则通过几何关系得到δ与反正切函数的关系:

$$\delta = \arctan\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right)$$

设板凳沿直线方向的半长 $d_1 = 0.275(m)$，板凳宽度方向的半宽为 $d_2 = 0.15(m)$。由于板凳是矩形，其顶点需要根据宽度和长度在垂直于和平行于直线 M_iM_{i+1} 的方向上偏移得到。接下来我们求沿着直线M_iM_{i+1}的单位向量u和垂直

于直线M_iM_{i+1}的n。

$$u=(cos\delta,sin\delta)$$
$$n=(sin\delta,-cos\delta)$$

顶点A_i，B_i，C_{ii}，D_i可以通过从M_i，M_{i+1}两点出发，沿着u和n两个向量得到。

$$\begin{cases}x_{Ai}=x_i+d_1cos\delta+d_2sin\delta\\y_{Ai}=y_i+d_1cos\delta-d_2cos\delta\end{cases}$$
$$\begin{cases}x_{Bi}=x_i+d_1cos\delta-d_2sin\delta\\y_{Bi}=y_i+d_1cos\delta+d_2cos\delta\end{cases}$$
$$\begin{cases}x_{Ci}=x_{i+1}-d_1cos\delta+d_2sin\delta\\y_{Ci}=y_{i+1}-d_1cos\delta-d_2sin\delta\end{cases}$$
$$\begin{cases}x_{Di}=x_{i+1}-d_1cos\delta-d_2sin\delta\\y_{Di}=y_{i+1}-d_1cos\delta+d_2cos\delta\end{cases}$$

其中$d_1=0.275(m)$，$d_2=0.15(m)$，$\delta=\arctan\left(\frac{y_{i+1}-y_i}{x_{i+1}-x_i}\right)$。

为探求相邻板是否发生碰撞，则要让第i节板的各边不与第$i+1$节板有交点，所以通过上述求来的坐标值，我们可以写出各节板上的线段方程。例如AB线段

$$\begin{cases}x=x_A+p_ih\\y=y_A+g_ih\end{cases}\quad h\epsilon[0,1]，p_i=x_B-x_A，g_i=y_B-y_A$$

与另外一节板上的 EF 线段：

$$\begin{cases}x=x_E+p_is\\y=y_E+g_is\end{cases}\quad s\epsilon[0,1]，p_j=x_F-x_E，g_j=y_F-y_E$$

两线段相交：

$$\begin{cases}x_A+p_ih=x_E+p_is\\y_A+g_ih=y_E+g_is\end{cases}\overset{转换}{\Longleftrightarrow}\begin{cases}p_ih-p_js=x_E-x_A\\g_ih-g_js=y_E-y_A\end{cases}$$
$$\underset{推出}{\Longrightarrow}\begin{cases}s=\dfrac{p_i(y_E-y_A)-g_i(x_E-x_A)}{p_jg_i-p_ig_j}\\h=\dfrac{p_j(y_E-y_A)-g_j(x_E-x_A)}{p_jg_i-p_ig_j}\end{cases}$$

为方便书写，令$\nabla=p_jg_i-p_ig_j$，因为其在分母上，所以要进行对∇的分类讨论，分为$\nabla\neq 0$ 时，且$0\le h\le 1$和$0\le s\le 1$同时满足时，发生碰撞，否则不发生碰撞。当$\nabla=0$时,两线段可能发生重合，或者平行，根据我们的假设恰好发

生重合不算做发生碰撞，因此不发生碰撞。一个节板需要做十六次这样的循环，来判断相邻板凳是否发生碰撞。如果不相互碰撞，则以当前步长增加时间，继续执行。若满足碰撞条件时，缩短步长，继续执行，当得到七位小数时，四舍五入保留六位小数，得到最终结束时间。

问题三：调头区域的求解

结合模型二的相关数据和题目给的信息，我们要选找一个最小螺距。结合题目给的直径为 9m 的调头区域设定初始步长，慢慢缩小螺距，判断是否发生碰撞，若发生碰撞，返回上一步，减小步长继续执行，逐渐逼近最小螺距。

对于螺距的初始设定，我们先以 0.55m 为初始值，计算出此时龙头前把手的极径为 m,对比发现小于调头空间的半径 4.5m，则说明在设定内可以不发生碰撞成功舞龙，因此 0.55m 可以满足我们螺距的初始设定，接下来我们从该值开始减小，直到找到最小螺距。

附录

问题一

```
a = 0.55;         % 螺距(m)
v0 = 1.0;         % 龙头恒定速度(m/s)
R1 = 2.86;        % 龙头把手间距(m)
R2 = 1.65;        % 龙身和龙尾把手间距(m)
theta0 = 16*2*pi; % 初始极角
n = 224;          % 把手总数
```

```
dt = 1;         % 时间步长(s)
t_total = 300;  % 总时间(s)

% 1. 精确求解龙头轨迹
t = 0:dt:t_total;

ode_fun = @(t,th) (2*pi*v0)/(a*sqrt(1+th^2));
[~, theta_ode] = ode45(ode_fun, t, theta0, odeset('RelTol',1e-9));
theta = zeros(n, length(t));
theta(1,:) = theta_ode';

% 2. 改进的把手位置求解
spacing_eqn = @(th2, th1, R) (a/(2*pi))^2*(th2^2 + th1^2 -
2*th1*th2*cos(th2-th1)) - R^2;
options = optimset('TolX',1e-8,'Display','off');

fprintf('计算把手位置...进度: ');
progress_step = floor(length(t)/10);

for i = 1:length(t)
    % 显示进度
    if mod(i, progress_step) == 0
        fprintf('%d%% ', round(i/length(t)*100));
    end

    % 龙头后把手
    if i == 1
        th_guess = theta(1,1) - 2*pi*R1/a;
    else
        th_guess = theta(2,i-1);
    end

    search_range = [max(1e-6, th_guess-0.5), th_guess+0.5];
    f_lower = spacing_eqn(search_range(1), theta(1,i), R1);
    f_upper = spacing_eqn(search_range(2), theta(1,i), R1);

    if sign(f_lower) == sign(f_upper)
        search_range = [max(1e-6, theta(1,i)-4*pi*R1/a), theta(1,i)];
    end

    try
        [theta(2,i), ~, exitflag] = fzero(@(th) spacing_eqn(th, theta(1,i),
R1), ...
                                  search_range, options);
```

```
        if exitflag ~= 1
            error('fzero 未收敛');
        end
    catch
        theta(2,i) = theta(1,i) - 2*pi*R1/a;
        warning('时间%d: 使用近似解计算龙头后把手位置', t(i));
    end

    % 后续
    for j = 3:n
        if i == 1
            th_guess = theta(j-1,1) - 2*pi*R2/a;
        else
            th_guess = theta(j,i-1);
        end

        search_range = [max(1e-6, th_guess-0.3), th_guess+0.3];
        f_lower = spacing_eqn(search_range(1), theta(j-1,i), R2);
        f_upper = spacing_eqn(search_range(2), theta(j-1,i), R2);

        if sign(f_lower) == sign(f_upper)
            search_range = [max(1e-6, theta(j-1,i)-4*pi*R2/a), theta(j-
1,i)];
        end

        try
            theta(j,i) = fzero(@(th) spacing_eqn(th, theta(j-1,i), R2), ...
                                search_range, options);
        catch
            theta(j,i) = theta(j-1,i) - 2*pi*R2/a;
            if mod(j,20) == 0
                warning('时间%d 把手%d: 使用近似解', t(i), j);
            end
        end
    end
end
fprintf('\n');

% 3. 计算位置和速度
r = (a/(2*pi)) * theta;  % 极径
x = r .* cos(theta);     % x 坐标
y = r .* sin(theta);     % y 坐标

% 速度计算（中心差分法）
```

```
v = zeros(size(theta));
for i = 2:length(t)-1
    for j = 1:n
        dx = (x(j,i+1) - x(j,i-1))/(2*dt);
        dy = (y(j,i+1) - y(j,i-1))/(2*dt);
        v(j,i) = sqrt(dx^2 + dy^2);
    end
end
v(:,1) = v(:,2);       % 前向差分处理首点
v(:,end) = v(:,end-1); % 后向差分处理末点

%龙头速度精确为1.000000
v(1,:) = 1.000000;

% 4. 生成特定时间点的位置和速度表格
output_times = [0, 60, 120, 180, 240, 300];
output_handles = [1, 2, 52, 102, 152, 202, 224]; % 龙头前、龙头后第1、51、101、151、201节、龙尾后

% 初始化结果表格
pos_table = cell(length(output_times)+1, length(output_handles)*2 + 1);
vel_table = cell(length(output_times)+1, length(output_handles) + 1);

% 设置表头
pos_table{1,1} = '时间(s)';
vel_table{1,1} = '时间(s)';

for h = 1:length(output_handles)
    handle_idx = output_handles(h);
    if handle_idx == 1
        name = '龙头前';
    elseif handle_idx == 224
        name = '龙尾后';
    else
        name = sprintf('第%d节', handle_idx-1);
    end

    % 位置表头
    pos_table{1, h*2} = [name ' x'];
    pos_table{1, h*2+1} = [name ' y'];

    % 速度表头
    vel_table{1, h+1} = name;
end
```

```matlab
% 填充数据
for i = 1:length(output_times)
    t_idx = output_times(i) + 1; % 因为时间从 0 开始，MATLAB 索引从 1 开始

    % 时间列
    pos_table{i+1,1} = output_times(i);
    vel_table{i+1,1} = output_times(i);

    for h = 1:length(output_handles)
        handle_idx = output_handles(h);

        % 位置数据 (x,y)
        pos_table{i+1, h*2} = round(x(handle_idx, t_idx), 6);
        pos_table{i+1, h*2+1} = round(y(handle_idx, t_idx), 6);

        % 速度数据
        vel_table{i+1, h+1} = round(v(handle_idx, t_idx), 6);
    end
end

% 显示表格
disp('特定时间点的位置数据(x,y 坐标，单位:m):');
disp(cell2table(pos_table(2:end,:), 'VariableNames', pos_table(1,:)));

disp('特定时间点的速度数据(单位:m/s):');
disp(cell2table(vel_table(2:end,:), 'VariableNames', vel_table(1,:)));

% 将结果写入 Excel
output_filename = 'result1.xlsx';

% 写入位置数据
writetable(cell2table(pos_table(2:end,:), 'VariableNames',
pos_table(1,:)), ...
         output_filename, 'Sheet', '位置数据');

% 写入速度数据
writetable(cell2table(vel_table(2:end,:), 'VariableNames',
vel_table(1,:)), ...
         output_filename, 'Sheet', '速度数据');

fprintf('\n 结果已保存到%s\n', output_filename);

% 5. 可视化
```

```
figure;
set(gcf,'Position',[100 100 1200 600]);

% 轨迹图
subplot(1,2,1);
hold on;
colors = lines(length(output_handles));
legend_names = cell(1, length(output_handles));

for k = 1:length(output_handles)
    h = output_handles(k);
    if h == 1
        legend_names{k} = '龙头前';
    elseif h == 224
        legend_names{k} = '龙尾后';
    else
        legend_names{k} = sprintf('第%d 节', h-1);
    end

    plot(x(h,:), y(h,:), 'Color', colors(k,:), 'LineWidth', 1.5, ...
        'DisplayName', legend_names{k});
end
title('关键把手运动轨迹');
xlabel('x (m)'); ylabel('y (m)');
legend('Location', 'bestoutside');
axis equal; grid on;

% 速度衰减曲线
subplot(1,2,2);
plot(t, v(output_handles,:), 'LineWidth', 2);
legend(legend_names, 'Location', 'eastoutside');
title('各把手速度随时间变化');
xlabel('时间(s)'); ylabel('速度(m/s)');
grid on;
ylim([0.996, 1.001]);
```

问题二：

```
function [isCollision] = checkCollision(xi, yi, xi_plus1, yi_plus1,
xi_plus2, yi_plus2)
    % 参数定义
    d1 = 0.275; % 板凳沿直线方向的半长
    d2 = 0.15;  % 板凳宽度方向的半宽
```

```
    % 计算夹角 delta
    delta_i = atan2(yi_plus1 - yi, xi_plus1 - xi);
    delta_i_plus1 = atan2(yi_plus2 - yi_plus1, xi_plus2 - xi_plus1);

    % 计算第 i 节板的四个顶点坐标
    [xA_i, yA_i, xB_i, yB_i, xC_i, yC_i, xD_i, yD_i] =
calculateVertices(xi_plus1, yi_plus1, delta_i, d1, d2);

    % 计算第 i+1 节板的四个顶点坐标
    [xA_ip1, yA_ip1, xB_ip1, yB_ip1, xC_ip1, yC_ip1, xD_ip1, yD_ip1] =
calculateVertices(xi_plus2, yi_plus2, delta_i_plus1, d1, d2);

    % 定义第 i 节板的四条边
    edges_i = [xA_i, yA_i, xB_i, yB_i;  % AB
               xB_i, yB_i, xD_i, yD_i;  % BD
               xD_i, yD_i, xC_i, yC_i;  % DC
               xC_i, yC_i, xA_i, yA_i]; % CA

    % 定义第 i+1 节板的四条边
    edges_ip1 = [xA_ip1, yA_ip1, xB_ip1, yB_ip1;  % AB
                 xB_ip1, yB_ip1, xD_ip1, yD_ip1;  % BD
                 xD_ip1, yD_ip1, xC_ip1, yC_ip1;  % DC
                 xC_ip1, yC_ip1, xA_ip1, yA_ip1]; % CA

    % 检查所有边对是否相交
    isCollision = false;
    for k = 1:4
        for l = 1:4
            % 提取边 k 和边 l 的坐标
            x1 = edges_i(k, 1); y1 = edges_i(k, 2);
            x2 = edges_i(k, 3); y2 = edges_i(k, 4);

            x3 = edges_ip1(l, 1); y3 = edges_ip1(l, 2);
            x4 = edges_ip1(l, 3); y4 = edges_ip1(l, 4);

            % 检查边 k 和边 l 是否相交
            if doLinesIntersect(x1, y1, x2, y2, x3, y3, x4, y4)
                isCollision = true;
                return;
            end
        end
    end
end
```

```
function [xA, yA, xB, yB, xC, yC, xD, yD] = calculateVertices(x, y, delta,
d1, d2)
    % 计算单位向量 u 和 n
    u = [cos(delta), sin(delta)];
    n = [sin(delta), -cos(delta)];

    % 计算四个顶点坐标
    xA = x + d1 * u(1) + d2 * n(1);
    yA = y + d1 * u(2) + d2 * n(2);

    xB = x + d1 * u(1) - d2 * n(1);
    yB = y + d1 * u(2) - d2 * n(2);

    xC = x - d1 * u(1) - d2 * n(1);
    yC = y - d1 * u(2) - d2 * n(2);

    xD = x - d1 * u(1) + d2 * n(1);
    yD = y - d1 * u(2) + d2 * n(2);
end

function [intersect] = doLinesIntersect(x1, y1, x2, y2, x3, y3, x4, y4)
    % 计算分母
    denom = (x4 - x3) * (y2 - y1) - (x2 - x1) * (y4 - y3);

    % 如果分母为0，线段平行或重合
    if denom == 0
        intersect = false;
        return;
    end

    % 计算参数 h 和 s
    h = ((x4 - x3) * (y3 - y1) - (y4 - y3) * (x3 - x1)) / denom;
    s = ((x2 - x1) * (y3 - y1) - (y2 - y1) * (x3 - x1)) / denom;

    % 检查 h 和 s 是否在[0,1]范围内
    if h >= 0 && h <= 1 && s >= 0 && s <= 1
        intersect = true;
    else
        intersect = false;
    end
end
```

问题三：