1. 空间 3R 肘机械臂各齐次变换矩阵有：

$$
{}^0T_1 = \begin{bmatrix} \cos\theta_1 & 0 & -\sin\theta_1 & 0 \\ \sin\theta_1 & 0 & \cos\theta_1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^1T_2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^2T_3 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & a_3\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

故位置级正运动学方程为：

$$
{}^0T_3 = {}^0T_1 \cdot {}^1T_2 \cdot {}^2T_3 = \begin{bmatrix} \cos\theta_1\cos(\theta_2+\theta_3) & -\cos\theta_1\sin(\theta_2+\theta_3) & -\sin\theta_1 & x \\ \sin\theta_1\cos(\theta_2+\theta_3) & -\sin\theta_1\sin(\theta_2+\theta_3) & \cos\theta_1 & y \\ -\sin(\theta_2+\theta_3) & -\cos(\theta_2+\theta_3) & 0 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

其中位置向量为：

$$
{}^0P_3 = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta_1(a_2\cos\theta_2 + a_3\cos(\theta_2+\theta_3)) \\ \sin\theta_1(a_2\cos\theta_2 + a_3\cos(\theta_2+\theta_3)) \\ d_1 - a_2\sin\theta_2 - a_3\sin(\theta_2+\theta_3) \end{bmatrix}
$$

下面计算圆轨迹参数方程：

由圆心及轨迹上点的坐标计算圆半径为：

$$
r = \|P_0 - O_c\| = \sqrt{0.2^2 + 0.1^2 + 0.2^2} = 0.3
$$

构建圆弧的局部坐标系。设 x 沿 $P_0 - O_c$ 方向：

$$i = \frac{P_0 - O_c}{\|P_0 - O_c\|} = \begin{bmatrix} 0.6667 \\ 0.3333 \\ 0.6667 \end{bmatrix}$$

为确定圆弧平面上的 j 矢量，利用 P_f 构造圆弧平面的法向量：

$$n = (P_0 - O_c) \times (P_f - O_c) = \begin{bmatrix} 0.2 \\ 0.1 \\ 0.2 \end{bmatrix} \times \begin{bmatrix} 0.1 \\ -0.2 \\ -0.2 \end{bmatrix} = \begin{bmatrix} 0.02 \\ 0.06 \\ -0.05 \end{bmatrix}$$

归一化得到 k 矢量（垂直于圆弧平面）：

$$k = \frac{n}{\|n\|} = \frac{1}{\sqrt{0.02^2 + 0.06^2 + 0.05^2}} \begin{bmatrix} 0.02 \\ 0.06 \\ -0.05 \end{bmatrix} = \begin{bmatrix} 0.2481 \\ 0.7442 \\ -0.6202 \end{bmatrix}$$

最后由右手系法则得：

$$j = k \times i = \begin{bmatrix} 0.7029 \\ -0.5788 \\ -0.4134 \end{bmatrix}$$

从 P_0 到 P_f 的圆心角为：

$$\phi_f = \arccos\left(\frac{(P_0 - O_c) \cdot (P_f - O_c)}{\|P_0 - O_c\| \|P_f - O_c\|}\right) = 116.39^\circ$$

圆轨迹参数方程为：

$$P(\lambda) = O_c + r(\cos(\phi_0 + \lambda(\phi_f - \phi_0)) \cdot i + \sin(\phi_0 + \lambda(\phi_f - \phi_0)) \cdot j) \quad \lambda \in [0, 1]$$

下面将参数时序化：

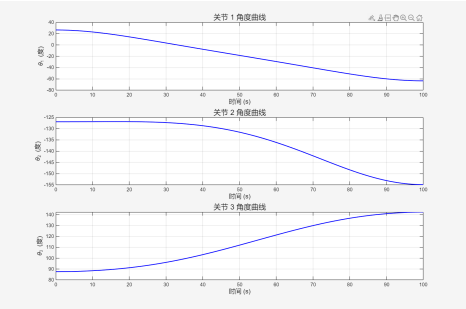
采用三次多项式进行时间规划：

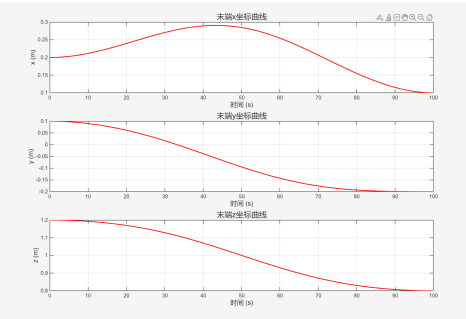
$$\lambda(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

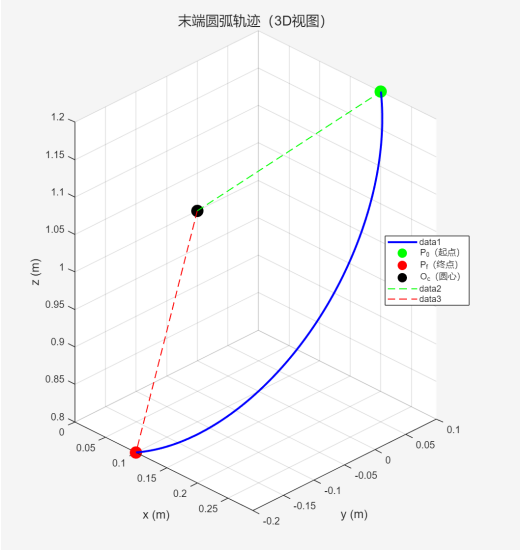
满足边界条件：$\lambda(0) = 0$，$\dot{\lambda}(0) = 0$，$\lambda(t_f) = 1$，$\dot{\lambda}(t_f) = 0$。确定为：

$$\lambda(t) = 3\left(\frac{t}{t_f}\right)^2 - 2\left(\frac{t}{t_f}\right)^3$$

求解得机器人关节角，末端位置曲线及 3D 轨迹如下：







附 Matlab 程序：

```
clear; clc; close all;

%% 参数设置
```

```
% 机械臂DH参数
d1 = 0.5; % m
a2 = 0.4; % m
a3 = 0.6; % m

% 关节初值
theta0_deg = [26.5651; -126.9498; 87.6120];
theta0 = deg2rad(theta0_deg); % 转换为弧度

% 关键点坐标
p0 = [0.2; 0.1; 1.2]; % 起点 P_0
pf = [0.1; -0.2; 0.8]; % 终点 P_f
Oc = [0; 0; 1]; % 圆心 O_c

% 时间参数
tf = 100; % 总时间 100s
dt = 0.1; % 采样周期 0.1s
t = 0:dt:tf; % 时间序列
N = length(t); % 采样点数



%% 圆弧轨迹规划
% 圆弧半径
r = norm(p0 - Oc);
fprintf('圆弧半径 r = %.4f m\n\n', r);

% 构建圆平面坐标系基向量 i, j, k
i_vec = (p0 - Oc) / norm(p0 - Oc);
fprintf('i = [%.4f, %.4f, %.4f]^T\n', i_vec(1), i_vec(2), i_vec(3));

v0 = p0 - Oc;
vf = pf - Oc;
n = cross(v0, vf);
fprintf('法向量 n = [%.4f, %.4f, %.4f]^T\n', n(1), n(2), n(3));

k_vec = n / norm(n);
fprintf('k = [%.4f, %.4f, %.4f]^T\n', k_vec(1), k_vec(2), k_vec(3));

j_vec = cross(k_vec, i_vec);
fprintf('j = [%.4f, %.4f, %.4f]^T\n', j_vec(1), j_vec(2), j_vec(3));

% 从 P_0 到 P_f 的圆心角
cos_phi_f = dot(v0, vf) / (norm(v0) * norm(vf));
phi_f = acos(cos_phi_f);
fprintf('P_0 到 P_f 的圆心角 phi_f = %.2f度\n', rad2deg(phi_f));



%% 时间规划（三次多项式）
tau = t / tf; % 归一化时间 [0,1]
lambda = 3*tau.^2 - 2*tau.^3;
lambda_dot = (6*tau - 6*tau.^2) / tf;
lambda_ddot = (6 - 12*tau) / tf^2;



%% 生成圆弧轨迹
p_traj = zeros(3, N);
```

```
63
64  phi_0 = 0; % 起点对应角度为 0
65  for i = 1:N
66     % 当前角度
67     phi = phi_0 + lambda(i) * (phi_f - phi_0);
68     % 圆弧轨迹
69     p_traj(:, i) = Oc + r * (cos(phi) * i_vec + sin(phi) * j_vec);
70  end
71
72
73
74  %% 逆运动学求解
75  theta_traj = zeros(3, N);
76
77  % 定义目标函数: 最小化正运动学误差的平方和
78  objective = @(theta, p_target) norm(forward_kinematics(theta, d1, a2, a3) - p_target)^2;
79
80  % 设置求解器选项
81  options = optimset(...
82     'Display', 'off', ... % 不显示迭代信息
83     'TolFun', 1e-12, ... % 函数容差
84     'TolX', 1e-10, ... % 变量容差
85     'MaxIter', 1000, ... % 最大迭代次数
86     'MaxFunEvals', 3000); % 最大函数评估次数
87
88  fprintf('\n开始逆运动学求解...\n');
89
90  for i = 1:N
91     p_target = p_traj(:, i);
92
93     % 使用前一时刻的解作为初值 (第一次使用给定初值)
94     if i == 1
95        theta_init = theta0;
96     else
97        theta_init = theta_traj(:, i-1);
98     end
99
100    % 使用 fminsearch 求解 (最小化误差)
101    [theta_sol, fval, exitflag] = fminsearch(@(theta) objective(theta, p_target), ...
102                                    theta_init, options);
103
104    % 检查求解状态和精度
105    if exitflag <= 0
106       warning('在时刻 t=%.2f 处逆运动学求解未收敛, exitflag=%d', t(i), exitflag);
107    end
108    if fval > 1e-6
109       warning('在时刻 t=%.2f 处逆运动学求解精度较低, 残差=%.6f', t(i), sqrt(fval));
110    end
111
112    theta_traj(:, i) = theta_sol;
113
114    % 每100个点显示一次进度
115    if mod(i, 100) == 0
116       fprintf('已完成 %d/%d 点 (%.1f%%)\n', i, N, 100*i/N);
117    end
118 end
119
120 fprintf('逆运动学求解完成! \n');
121
```

```
% 转换为角度
theta_traj_deg = rad2deg(theta_traj);



%% 验证正运动学
% 验证起点的正运动学
p0_verify = forward_kinematics(theta0, d1, a2, a3);
fprintf('\n正运动学验证（初始关节角对应位置）：\n');
fprintf('关节角: theta = [%.4f, %.4f, %.4f]^T (度)\n', theta0_deg(1), theta0_deg(2), theta0_deg(3));
fprintf('给定起点: P_0 = [%.4f, %.4f, %.4f]^T\n', p0(1), p0(2), p0(3));
fprintf('正运动学计算: P = [%.4f, %.4f, %.4f]^T\n', p0_verify(1), p0_verify(2), p0_verify(3));
fprintf('误差: %.6f m\n', norm(p0 - p0_verify));



%% 绘图
% 关节角曲线
figure('Name', '关节角曲线', 'Position', [100, 100, 1200, 800]);
for i = 1:3
    subplot(3, 1, i);
    plot(t, theta_traj_deg(i, :), 'b-', 'LineWidth', 1.5);
    grid on;
    xlabel('时间 (s)', 'FontSize', 12);
    ylabel(['\theta_' num2str(i) ' (度)'], 'FontSize', 12);
    title(['关节 ' num2str(i) ' 角度曲线'], 'FontSize', 14);
    xlim([0, tf]);
end

% 末端位置曲线
figure('Name', '末端位置曲线', 'Position', [150, 150, 1200, 800]);
coords = {'x', 'y', 'z'};
for i = 1:3
    subplot(3, 1, i);
    plot(t, p_traj(i, :), 'r-', 'LineWidth', 1.5);
    grid on;
    xlabel('时间 (s)', 'FontSize', 12);
    ylabel([coords{i} ' (m)'], 'FontSize', 12);
    title(['末端' coords{i} '坐标曲线'], 'FontSize', 14);
    xlim([0, tf]);
end

% 3D轨迹
figure('Name', '3D轨迹', 'Position', [200, 200, 800, 800]);
plot3(p_traj(1, :), p_traj(2, :), p_traj(3, :), 'b-', 'LineWidth', 2);
hold on;

% 标记关键点
plot3(p0(1), p0(2), p0(3), 'go', 'MarkerSize', 12, 'MarkerFaceColor', 'g', 'DisplayName', 'P_0 (起点) ');
plot3(pf(1), pf(2), pf(3), 'ro', 'MarkerSize', 12, 'MarkerFaceColor', 'r', 'DisplayName', 'P_f (终点) ');
plot3(Oc(1), Oc(2), Oc(3), 'ko', 'MarkerSize', 12, 'MarkerFaceColor', 'k', 'DisplayName', 'O_c (圆心) ');

% 绘制从圆心到关键点的连线
plot3([Oc(1), p0(1)], [Oc(2), p0(2)], [Oc(3), p0(3)], 'g--', 'LineWidth', 1);
plot3([Oc(1), pf(1)], [Oc(2), pf(2)], [Oc(3), pf(3)], 'r--', 'LineWidth', 1);

grid on;
xlabel('x (m)', 'FontSize', 12);
ylabel('y (m)', 'FontSize', 12);
```

```
zlabel('z (m)', 'FontSize', 12);
title('末端圆弧轨迹（3D视图）', 'FontSize', 14);
legend('Location', 'best');
axis equal;
view(45, 30);



%% 辅助函数
% 正运动学函数
function p = forward_kinematics(theta, d1, a2, a3)
   theta1 = theta(1);
   theta2 = theta(2);
   theta3 = theta(3);

   x = cos(theta1) * (a2*cos(theta2) + a3*cos(theta2+theta3));
   y = sin(theta1) * (a2*cos(theta2) + a3*cos(theta2+theta3));
   z = d1 - a2*sin(theta2) - a3*sin(theta2+theta3);

   p = [x; y; z];
end
```

2. 空间 3R 球腕机械臂各齐次变换矩阵有：

$$
{}^0T_1 = \begin{bmatrix} \cos\theta_1 & 0 & -\sin\theta_1 & 0 \\ \sin\theta_1 & 0 & \cos\theta_1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^1T_2 = \begin{bmatrix} \cos\theta_2 & 0 & \sin\theta_2 & 0 \\ \sin\theta_2 & 0 & -\cos\theta_2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^2T_3 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

故姿态级正运动学方程为：

$$
{}^0T_3 = {}^0T_1 \cdot {}^1T_2 \cdot {}^2T_3 = \begin{bmatrix} {}^0R_3 & {}^0P_3 \\ 0 & 1 \end{bmatrix}
$$

其中旋转矩阵为：

$$
{}^0R_3 = \begin{bmatrix} \cos\theta_1\cos\theta_2\cos\theta_3 + \sin\theta_1\sin\theta_3 & -\cos\theta_1\cos\theta_2\sin\theta_3 + \sin\theta_1\cos\theta_3 & -\cos\theta_1\sin\theta_2 \\ \sin\theta_1\cos\theta_2\cos\theta_3 - \cos\theta_1\sin\theta_3 & -\sin\theta_1\cos\theta_2\sin\theta_3 - \cos\theta_1\cos\theta_3 & -\sin\theta_1\sin\theta_2 \\ -\sin\theta_2\cos\theta_3 & \sin\theta_2\sin\theta_3 & \cos\theta_2 \end{bmatrix}
$$

位置向量为：

$$
{}^0P_3 = \begin{bmatrix} -d_3\cos\theta_1\sin\theta_2 \\ -d_3\sin\theta_1\sin\theta_2 \\ d_1 + d_3\cos\theta_2 \end{bmatrix}
$$

(1) 采用三次多项式进行时间规划：

$$
s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3
$$

满足边界条件：$s(0)=0$，$\dot{s}(0)=0$，$s(t_f)=1$，$\dot{s}(t_f)=0$，其中 $t_f=10$ s。求解得：

$$\begin{cases} a_0 = 0 \\ a_1 = 0 \\ a_2 = \frac{3}{t_f^2} \\ a_3 = -\frac{2}{t_f^3} \end{cases}$$

因此归一化参数为：

$$s(t) = 3\left(\frac{t}{t_f}\right)^2 - 2\left(\frac{t}{t_f}\right)^3$$

对姿态角各分量进行插值：

$$\Phi(t) = \Phi_0 + (\Phi_f - \Phi_0) \cdot s(t)$$

其中初始姿态角 $\Phi_0 = [-157.8240^\circ, 46.0418^\circ, 70.4798^\circ]^T$，终止姿态角 $\Phi_f = [174.9616^\circ, 8.6492^\circ, 90.3813^\circ]^T$。

(2) 首先计算初始和终止姿态对应的旋转矩阵。xyz 动轴欧拉角对应的旋转矩阵为：

$$R = R_x(\phi)R_y(\theta)R_z(\psi)$$

计算得：

$$R_0 = R_x(-157.8240^\circ)R_y(46.0418^\circ)R_z(70.4798^\circ)$$

$$R_f = R_x(174.9616^\circ)R_y(8.6492^\circ)R_z(90.3813^\circ)$$

计算相对旋转矩阵：

$$R_{rel} = R_0^T R_f$$

从旋转矩阵转换至轴-角有：

$$\theta_{eq} = \arccos\left(\frac{\mathrm{tr}(R_{rel}) - 1}{2}\right)$$

$$\mathbf{k} = \frac{1}{2\sin\theta_{eq}} \begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix}$$

采用五次多项式进行时间规划：

$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

满足边界条件：$s(0) = 0$, $\dot{s}(0) = 0$, $\ddot{s}(0) = 0$, $s(t_f) = 1$, $\dot{s}(t_f) = 0$, $\ddot{s}(t_f) = 0$。求解得：

$$\begin{cases} a_0 = a_1 = a_2 = 0 \\ a_3 = \frac{10}{t_f^3} \\ a_4 = -\frac{15}{t_f^4} \\ a_5 = \frac{6}{t_f^5} \end{cases}$$

因此：

$$s(t) = 10\left(\frac{t}{t_f}\right)^3 - 15\left(\frac{t}{t_f}\right)^4 + 6\left(\frac{t}{t_f}\right)^5$$

对等效转角进行插值：

$$\theta(t) = \theta_{eq} \cdot s(t)$$

则每个 t 时刻相对旋转矩阵为：

$$R(t) = I + \sin\theta(t) \cdot K + (1 - \cos\theta(t)) \cdot K^2$$

其中 K 为等效转轴 $\mathbf{k} = [k_x, k_y, k_z]^T$ 对应的反对称矩阵（斜对称矩阵），定义为：

$$K = \mathbf{k}^{\times} = \begin{bmatrix} 0 & -k_z & k_y \\ k_z & 0 & -k_x \\ -k_y & k_x & 0 \end{bmatrix}$$

两种规划方法对比如下：

方法 1（三次多项式直接插值）：

- 优点：实现简单，计算量小
- 缺点：各姿态角分量独立插值，可能导致非最短路径旋转；角速度在端点虽然为零，但中间可能不够平滑

方法 2（五次多项式等效转角插值）：

- 优点：基于等效转轴和转角，保证最短路径旋转；五次多项式使加速度在起止点为零，运动更加平滑；物理意义明确
- 缺点：计算稍复杂，需要旋转矩阵的转换

求解得到的姿态角曲线、角速度曲线及关节角曲线如下图所示。

附 Matlab 程序：

```
clear; clc; close all;

%% 参数设置
d1 = 0.3; % m
d3 = 0.5; % m

% 初始关节角
theta0_deg = [20; -50; 100];
theta0 = deg2rad(theta0_deg);

% 初始和终止姿态角（xyz动轴欧拉角）
phi0_deg = [-157.8240; 46.0418; 70.4798];
phif_deg = [174.9616; 8.6492; 90.3813];
phi0 = deg2rad(phi0_deg);
phif = deg2rad(phif_deg);

% 时间参数
tf = 10; % 总时间 10s
dt = 0.1; % 采样周期 0.1s
t = 0:dt:tf;
N = length(t);
```

```
%% 方法1
tau1 = t / tf;
s1 = 3*tau1.^2 - 2*tau1.^3;
s1_dot = (6*tau1 - 6*tau1.^2) / tf;

phi1_traj = phi0 + (phif - phi0) .* s1;
phi1_dot_traj = (phif - phi0) .* s1_dot;
phi1_traj_deg = rad2deg(phi1_traj);



%% 方法2)
R0 = euler_xyz_to_rotation(phi0);
Rf = euler_xyz_to_rotation(phif);
R_rel = R0' * Rf;
[theta_eq, k_axis] = rotation_to_axis_angle(R_rel);

tau2 = t / tf;
s2 = 10*tau2.^3 - 15*tau2.^4 + 6*tau2.^5;
theta_traj = theta_eq * s2;

phi2_traj = zeros(3, N);
phi2_dot_traj = zeros(3, N);

for i = 1:N
   R_current = axis_angle_to_rotation(k_axis, theta_traj(i));
   R_abs = R0 * R_current;
   phi2_traj(:, i) = rotation_to_euler_xyz(R_abs);

   if i > 1
      phi2_dot_traj(:, i) = (phi2_traj(:, i) - phi2_traj(:, i-1)) / dt;
   end
end
phi2_dot_traj(:, 1) = phi2_dot_traj(:, 2);
phi2_traj_deg = rad2deg(phi2_traj);



%% 计算关节角轨迹
theta1_traj = zeros(3, N);
for i = 1:N
   if i == 1
      theta_init = theta0;
   else
      theta_init = theta1_traj(:, i-1);
   end

   R_target = euler_xyz_to_rotation(phi1_traj(:, i));
   theta1_traj(:, i) = inverse_kinematics_wrist(R_target, d1, d3, theta_init);
end
theta1_traj_deg = rad2deg(theta1_traj);

theta2_traj = zeros(3, N);
for i = 1:N
   if i == 1
      theta_init = theta0;
   else
      theta_init = theta2_traj(:, i-1);
   end
```

```
84
85      R_target = euler_xyz_to_rotation(phi2_traj(:, i));
86      theta2_traj(:, i) = inverse_kinematics_wrist(R_target, d1, d3, theta_init);
87  end
88  theta2_traj_deg = rad2deg(theta2_traj);
89
90
91
92  %% 绘图
93  figure('Name', '末端姿态角变化曲线');
94  euler_labels = {'\phi (roll)', '\theta (pitch)', '\psi (yaw)'};
95  for i = 1:3
96      subplot(3, 1, i);
97      plot(t, phi1_traj_deg(i, :), 'b-', 'LineWidth', 1.5, 'DisplayName', '方法1');
98      hold on;
99      plot(t, phi2_traj_deg(i, :), 'r--', 'LineWidth', 1.5, 'DisplayName', '方法2');
100     grid on;
101     xlabel('时间 (s)');
102     ylabel([euler_labels{i} ' (度)']);
103     legend('Location', 'best');
104 end
105
106 figure('Name', '末端角速度曲线');
107 for i = 1:3
108     subplot(3, 1, i);
109     plot(t, phi1_dot_traj(i, :), 'b-', 'LineWidth', 1.5, 'DisplayName', '方法1');
110     hold on;
111     plot(t, phi2_dot_traj(i, :), 'r--', 'LineWidth', 1.5, 'DisplayName', '方法2');
112     grid on;
113     xlabel('时间 (s)');
114     ylabel(['\omega_' num2str(i) ' (rad/s)']);
115     legend('Location', 'best');
116 end
117
118 figure('Name', '关节角曲线');
119 for i = 1:3
120     subplot(3, 1, i);
121     plot(t, theta1_traj_deg(i, :), 'b-', 'LineWidth', 1.5, 'DisplayName', '方法1');
122     hold on;
123     plot(t, theta2_traj_deg(i, :), 'r--', 'LineWidth', 1.5, 'DisplayName', '方法2');
124     grid on;
125     xlabel('时间 (s)');
126     ylabel(['\theta_' num2str(i) ' (度)']);
127     legend('Location', 'best');
128 end
129
130 %% 辅助函数
131
132 % xyz动轴欧拉角转旋转矩阵（按 Rx * Ry * Rz 顺序）
133 function R = euler_xyz_to_rotation(phi)
134     alpha = phi(1); % 绕x轴
135     beta = phi(2); % 绕y轴
136     gamma = phi(3); % 绕z轴
137
138     Rx = [1, 0, 0;
139         0, cos(alpha), -sin(alpha);
140         0, sin(alpha), cos(alpha)];
141
142     Ry = [cos(beta), 0, sin(beta);
```

```
        0, 1, 0;
        -sin(beta), 0, cos(beta)];

    Rz = [cos(gamma), -sin(gamma), 0;
         sin(gamma), cos(gamma), 0;
         0, 0, 1];

    % 动轴欧拉角: 先绕固定轴X, 再绕新Y, 最后绕新Z
    R = Rx * Ry * Rz;
end

% 旋转矩阵转xyz动轴欧拉角
function phi = rotation_to_euler_xyz(R)
    % 从 R = Rx * Ry * Rz 提取欧拉角
    % R(3,2) = sin(alpha)*cos(beta)*sin(gamma) + cos(alpha)*cos(gamma)
    % R(2,3) = sin(alpha)*sin(gamma) + cos(alpha)*cos(beta)*cos(gamma)

    beta = asin(R(1,3)); % sin(beta) = R(1,3)

    if abs(cos(beta)) > 1e-6
        alpha = atan2(-R(2,3), R(3,3));
        gamma = atan2(-R(1,2), R(1,1));
    else
        % 万向锁情况
        alpha = 0;
        gamma = atan2(R(2,1), R(2,2));
    end

    phi = [alpha; beta; gamma];
end

% 旋转矩阵转等效轴角
function [theta, k] = rotation_to_axis_angle(R)
    theta = acos((trace(R) - 1) / 2);

    if abs(theta) < 1e-6
        k = [0; 0; 1];
        theta = 0;
    else
        k = (1 / (2*sin(theta))) * [R(3,2)-R(2,3); R(1,3)-R(3,1); R(2,1)-R(1,2)];
        k = k / norm(k);
    end
end

% 轴角转旋转矩阵 (罗德里格斯公式)
function R = axis_angle_to_rotation(k, theta)
    K = [0, -k(3), k(2);
         k(3), 0, -k(1);
         -k(2), k(1), 0];

    R = eye(3) + sin(theta)*K + (1-cos(theta))*(K*K);
end

% 球腕逆运动学
function theta = inverse_kinematics_wrist(R_target, d1, d3, theta_init)
    % 简化的球腕逆运动学 (基于数值优化)
    objective = @(th) norm(compute_wrist_rotation(th, d1, d3) - R_target, 'fro')^2;

    options = optimset('Display', 'off', 'TolFun', 1e-10, 'TolX', 1e-10, 'MaxIter', 500);
```

```
    theta = fminsearch(objective, theta_init, options);
end

% 计算球腕旋转矩阵
function R = compute_wrist_rotation(theta, d1, d3)
    % 3R球腕的旋转矩阵计算
    c1 = cos(theta(1)); s1 = sin(theta(1));
    c2 = cos(theta(2)); s2 = sin(theta(2));
    c3 = cos(theta(3)); s3 = sin(theta(3));

    % 简化的球腕旋转
    R = [c1*c2*c3-s1*s3, -c1*c2*s3-s1*c3, c1*s2;
        s1*c2*c3+c1*s3, -s1*c2*s3+c1*c3, s1*s2;
        -s2*c3, s2*s3, c2];
end
```