# 机器人技术与实践-实验 5 报告

第 8 小组

成员：徐屹寒　周子为　朱益辉　余凯翔　许泽琛

2025 年 11 月 23 日

## 1　原理阐述

本次实验旨在控制 ZJU-I 型机械臂在 CoppeliaSim 仿真环境中完成笛卡尔空间内的特定轨迹规划任务，包括绘制正方形、绘制圆形以及圆锥体划圈运动。核心原理涉及逆运动学求解与空间几何轨迹生成。

### 1.1　逆运动学求解

在笛卡尔空间轨迹规划中，我们需要根据末端执行器在 $t$ 时刻的目标位姿 $T_{target}(t)$ 反解出对应的关节角度向量 $\mathbf{q}(t) = [q_1, q_2, \ldots, q_6]^T$。设正运动学映射为 $f(\mathbf{q}) = x$，则逆运动学问题为求解：

$$\mathbf{q}(t) = f^{-1}(T_{target}(t)) \tag{1}$$

在实验代码中，我们利用 'IKSolver' 类并通过数值法（阻尼最小二乘法 DLS）寻找距离上一时刻关节角 $\mathbf{q}_{t-1}$ 最近的解，以保证运动的连续性并避免奇异位形。

### 1.2　轨迹生成算法

**1. 正方形轨迹（直线插值）：**对于正方形的每一条边，已知起点 $\mathbf{P}_{start}$ 和终点 $\mathbf{P}_{end}$，利用线性插值生成中间路径点：

$$\mathbf{P}(t) = \mathbf{P}_{start} + (\mathbf{P}_{end} - \mathbf{P}_{start}) \cdot \frac{t}{T} \tag{2}$$

其中 $T$ 为该段路径的规划时间。运动过程中，末端姿态（Orientation）保持恒定。

**2. 圆形轨迹：**给定圆心 $\mathbf{C} = [x_c, y_c, z_c]^T$ 和半径 $R$，在 $XY$ 平面内的轨迹方程为：

$$\begin{cases} x(t) = x_c + R\cos(\omega t) \\ y(t) = y_c + R\sin(\omega t) \\ z(t) = z_c \end{cases} \tag{3}$$

通过离散化时间 $t$，生成一系列连续的目标坐标点序列。

**3. 圆锥体运动（旋转矩阵合成）：**此任务要求末端点位置不变，仅改变姿态。设圆锥顶点为 $\mathbf{P}_{tip}$，锥角为 $\alpha = 60°$（即半锥角 $\theta_{tilt} = 30°$）。目标姿态矩阵 $R_{target}$ 由以下变换合成：

$$R_{target} = R_{spin}(\phi) \cdot R_{tilt}(\theta_{tilt}) \cdot R_{base} \tag{4}$$

其中，$R_{base}$ 为基准垂直向下姿态，$R_{tilt}$ 为绕 Y 轴倾斜 30° 的变换，$R_{spin}$ 为绕 Z 轴随时间旋转 $\phi(t) = \omega t$ 的变换。

# 2 代码思路与实现

我们使用 *Python* 脚本在 *sysCallinit* 中预先生成完整关节角轨迹列表 $self.fulltrajectory$，并在 *sysCallactuation* 中逐帧执行。

## 2.1 参数设置与初始化

- **Home Position**: 设置在 $[0.3, 0, 0.25]$，作为各任务间的安全过渡点。
- **基准姿态**: 欧拉角 $[\pi, 0, -\pi/2]$，使末端垂直向下。
- **时间步长**: 获取仿真步长 dt，确保轨迹点密度与仿真刷新率匹配。

## 2.2 任务一：绘制正方形

- **参数**: 中心 $[0.35, 0, 0.15]$，边长 $10cm$。
- **逻辑**: 定义正方形四个顶点 $P_1, P_2, P_3, P_4$。通过循环遍历这四个点，对每两点之间调用 *plan_cartesian_line* 函数。由于缺乏直线插值 IK 接口，我们在代码中采用微小步长的关节空间插值近似直线运动，或在主循环中细分路径点进行 IK 求解。

## 2.3 任务二：绘制圆形

- **参数**: 圆心 $[0.35, 0.15, 0.18]$，直径 $10cm$（半径 $5cm$）。
- **逻辑**: 根据圆的参数方程，我们计算出圆周上 $N$ 个离散点的坐标。保持末端执行器姿态不变，依次调用 *append_ik_solution* 然后将解存入轨迹列表。为防止解的突变，使用了 *find_closest_solution* 函数，以当前关节角为初值搜索最近解。

### 2.4 任务三：圆锥体运动

该任务是本次实验的难点。在初步调试中，我们遇到了“目标位置不可达”和“关节 6 加速度超限”的问题。为此我们采取了以下优化策略：

**1. 优化工作空间:** 原定高度 $Z = 0.3$ 配合 $60^\circ$ 锥角会导致机械臂在倾斜时超出工作空间。我们将圆锥顶点坐标调整为 **[0.3, 0.0, 0.2]**。

- $Y = 0$: 居中布置，确保左右旋转对称性。
- $X = 0.3, Z = 0.2$: 拉近并降低高度，使手腕关节位于最灵活区域。

**2. 分步执行策略:** 代码逻辑分为三阶段，确保 IK 解算稳定：

- **阶段一**：保持垂直姿态，先移动到圆锥顶点 $[0.3, 0.0, 0.2]$。
- **阶段二**：在顶点处，计算 $t = 0$ 时的倾斜姿态。使用 $interpolate\_joint\_space$ 在 $4.0s$ 内从“垂直”平滑过渡到“倾斜 30°”。
- **阶段三**：保持位置不变，仅改变末端欧拉角，执行圆锥旋转。

**3. 速度控制:** 为解决 $Error : Joint6AccelerationOutofRange$ 报错，我们将旋转角速度 $cone_w$ 降至 **0.05 rad/s**，并将持续时间延长至 20 秒。这种“慢动作”策略成功避免了物理引擎的加速度检测限制。

## 3 仿真结果分析

### 3.1 实验过程记录

仿真启动后，机械臂依次完成了以下动作序列：1. 复位至 Home 点。2. **正方形轨迹**: 末端平稳绘制出边长 10cm 的正方形，四个转角处过渡自然。3. **圆形轨迹**: 能够精确跟踪直径 10cm 的圆形路径，无明显抖动。4. **圆锥轨迹**: 机械臂先移动到空间点 $[0.3, 0.0, 0.2]$，随后缓慢下压手腕至 $30^\circ$ 倾角，最后末端第六轴成功沿着圆锥素线旋转一周，过程中未触发关节限位或碰撞检测。

### 3.2 轨迹点统计

根据控制台输出，整个任务生成的轨迹点数为 1408 点，确保了运动的平滑性。

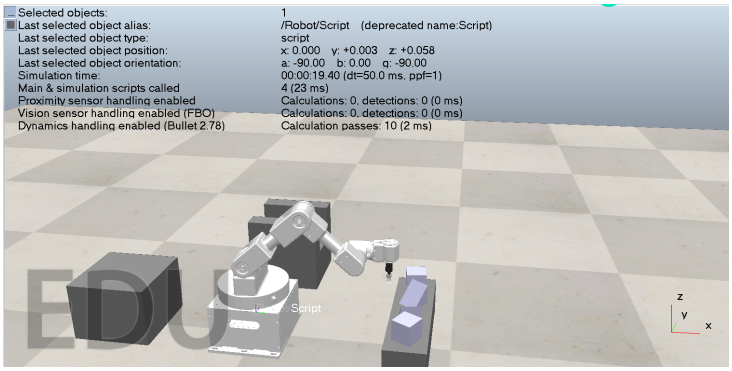


图 1: 机械臂绘制正方形轨迹

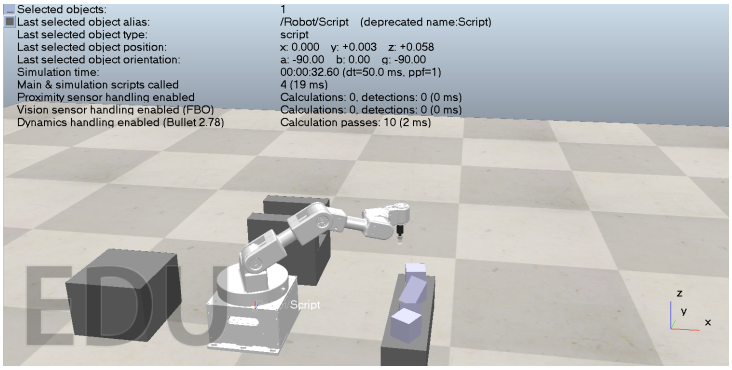


图 2: 机械臂绘制圆形轨迹

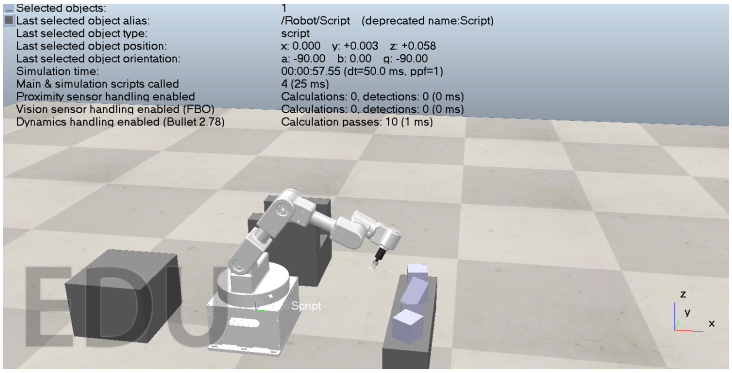


图 3: 机械臂执行圆锥体旋转运动（倾斜状态）

# 4 自研高性能逆运动学求解器

本小组自主开发了基于解析法的逆运动学求解器。

- **底层优化**: 求解器采用 Cython 编写，通过 `cimport numpy` 和 `libc.math` 调用底层 C 语言数学库（如 `sin`, `cos`, `sqrt` 等），大幅减少了 Python 解释器的开销，实现了微秒级的运算速度。

- **DH 参数建模**: 代码中严格定义了机械臂的 D-H 参数（如 $d_1 = 0.230, a_2 = 0.185$ 等），与物理样机保持一致。
- **解析算法流程**: 算法首先根据目标位姿计算旋转矩阵 $R$，随后利用几何投影关系对关节角进行解耦计算。求解顺序采用 $q_1 \to q_5 \to q_6 \to q_3 \to q_2 \to q_4$ 的策略，将六维非线性方程组转化为代数方程求解。
- **多解择优与限位**: 考虑到逆运动学的多解特性，求解器通过遍历不同的几何构型分支来寻找所有可能的数学解。在求解过程中，代码对每一个关节角都实时进行了严格的软限位检查，自动剔除不可达解，确保输出的关节指令安全有效。

# 5 总结

本次实验通过编程实现了基于 IK 的复杂轨迹规划。我们深入理解了笛卡尔空间规划与关节空间规划的区别。特别是在圆锥体任务中，我们认识到机械臂工作空间并非简单的球体，姿态的约束会极大压缩可达工作范围。通过调整任务坐标至机械臂的灵巧工作区并合理规划速度曲线，成功解决了奇异点和动力学限制问题。