三指灵巧手设计报告

罗凯耀[1]

(1. 广东工业大学 自动化学院，广东广州，510006)

摘 要 本报告详细介绍了三指灵巧手的设计过程，包括设计思路、正运动学、逆运动学、抓取规划、RRT算法以及机械臂装载等方面。通过对三指灵巧手的详细分析，展示了其设计原则、参数选择、运动学计算、抓取策略和路径规划等方面的研究成果。此外，报告还包含了实际的机械臂装载和演示，证明了设计的可行性和实用性。

关键词: 灵巧手；正逆运动学；抓取规划；路径规划；

中图法分类号: TP241 **文献标志码**: A

Design and Application of the Three-Fingered Dexterous Hand

Luo Kai-yao[1]

(1. School of Automation, Guangdong University and Technology, Guangzhou 510006, China)

Abstract The report delves into the intricate design journey of a three-fingered robotic hand, outlining the creative thought process, kinematic principles, both forward and inverse, the intricacies of grasp planning, the application of the RRT algorithm, and the assembly of the robotic arm. A comprehensive examination of the hand's design illuminates the guiding principles, decision-making for parameters, kinematic computations, and strategic grasping and routing. Moreover, the report features a hands-on assembly and demonstration of the robotic arm, substantiating the design's viability and real-world application.

Keywords Dexterous Hand; Forward and Inverse Kinematics; Grasp Planning; Path Planning;

随着机器人技术的不断发展，灵巧手作为机器人末端执行器的重要组成部分，在工业制造、医疗康复、航空航天等领域发挥着越来越重要的作用。灵巧手能够模拟人类手的复杂运动，实现精细操作和抓取，极大地拓展了机器人的应用范围。

然而，灵巧手的设计和运动控制仍然面临着诸多挑战，例如手指间的协调运动、关节的自由度、抓取点的规划、抓取力的控制等。为了解决这些问题，本研究设计了一种三指灵巧手，并对其运动学分析、抓取规划和机械臂装载等关键技术进行了深入研究。

本报告将详细介绍三指灵巧手的设计思路、正运动学模型、逆运动学求解方法、抓取与路径规划过程以及机械臂装载实例，旨在为灵巧手的设计和应用提供参考。

1 设计思路

1.1 设计原则

基于文献[1]，对人手的灵巧手的设计抽象出以下原则：

- 运动的协调性：确保手指间协调运动，例如手指同时张开或合拢，以实现平稳抓取和操作。
- 运动的独立性：屈伸运动和外展内收运动应独立，且手指的近端指间关节应独立于远端指间关节，以实现灵活的运动和抓取。
- 关节数量：尽可能增加关节数量，以提升灵活性和适应性，从而实现更复杂的抓取和操作。

1.2 设计参数

本研究设计的三指灵巧手如Figure 1,具有以下参数：

- 手指数量：3
- 手指自由度：4

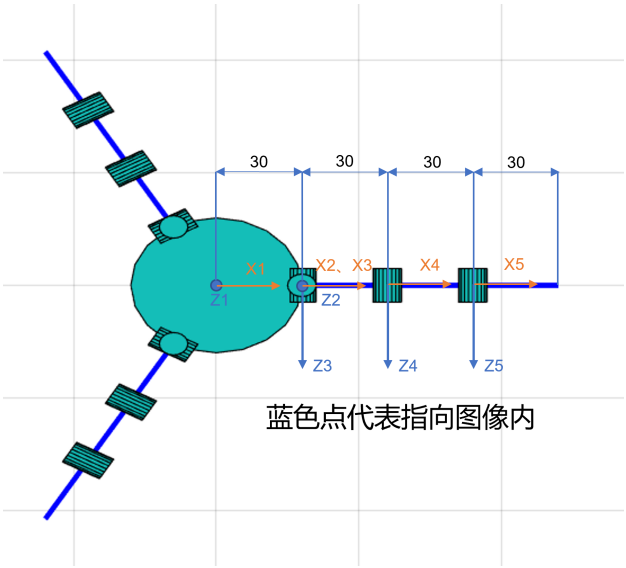

Fig. 1 坐标轴组图

- 每个手指有4个关节，包括一个外展/内收关节和一个屈伸关节。
- 手指长度：30-30-30-30（从掌心到指尖）
- 关节角度范围：
- 关节1（外展/内收）：-90°~90°
- 关节2（屈伸）：0°~60°
- 关节3（屈伸）：0°~60°
- 关节4（屈伸）：0°~80°

在此给出单根手指的标准DH表如Table 1。

Table 1 手指DH参数表

序号	θ	d	a	α	θ_i初值	运动范围
1	θ_1	0	30	0°	0°、±120°	-
2	θ_2	0	0	180°	0°	[-90°~90°]
3	θ_3	0	30	90°	0°	[0°~60°]
4	θ_4	0	30	0°	0°	[0°~60°]
5	θ_5	0	30	0°	0°	[0°~80°]

2 正逆运动学

2.1 正运动学

基于标准DH参数，建立了三指灵巧手的正运动学模型，并通过MATLAB实现了末端执行器位置的求解。正运动学模型描述了关节角度与末端执行器位置之间的映射关系，是进行逆运动学和抓取规划的基础。

关节n到关节$n+1$的坐标转换方程如Equation 1，可得手指各关节的齐次变换矩阵。

$$ {}^{n}T_{n+1} = \begin{bmatrix} C\theta_n & -C\alpha_n S\theta_n & S\alpha_n S\theta_n & a_n C\theta_n \\ S\theta_n & C\alpha_n C\theta_n & -S\alpha_n C\theta_n & a_n S\theta_n \\ 0 & S\alpha_n & C\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

$$ {}^{0}T_{1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2} $$

$$ {}^{1}T_{2} = \begin{pmatrix} \cos(\theta_1) & \sin(\theta_1) & 0 & 30\cos(\theta_1) \\ \sin(\theta_1) & -\cos(\theta_1) & 0 & 30\sin(\theta_1) \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3} $$

$$ {}^{2}T_{3} = \begin{pmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) & 0 \\ \sin(\theta_2) & 0 & -\cos(\theta_2) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4} $$

$$ {}^{3}T_{4} = \begin{pmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & 30\cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & 30\sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5} $$

$$ {}^{4}T_{5} = \begin{pmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 30\cos(\theta_4) \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 30\sin(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6} $$

$$ {}^{5}T_{end} = \begin{pmatrix} \cos(\theta_5) & -\sin(\theta_5) & 0 & 30\cos(\theta_5) \\ \sin(\theta_5) & \cos(\theta_5) & 0 & 30\sin(\theta_5) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{7} $$

自此，我们有：

$$ {}^{0}T_{end} = {}^{0}T_{1} * {}^{1}T_{2} * {}^{2}T_{3} * {}^{3}T_{4} * {}^{4}T_{5} * {}^{5}T_{end} \tag{8} $$

该矩阵较为庞大，在附件代码的*syms_output.mlx*中通过符号功能计算得到。

程序运行则实际通过附件*DHfk*文件夹中的代码实现。

2.2 逆运动学

为了实现从末端执行器位置到关节角度的映射，研究了逆运动学的数值解方法。逆运动学数值解是一种通过迭代方法求解机器人末端执行器位置到关节角度映射的方法。它利用雅可比矩阵描述机器人操作空间速度与关节空间速度之间的线性映射关系，并通过迭代更新关节角度，直至达到预设的末端执行器位置。

2.2.1 雅可比矩阵的微分变换法求解

雅可比矩阵*J*即可以看成从关节空间向操作空间速度传递的线性关系，也可以看成是微分运动转换的线性关系。

雅可比矩阵的微分变换法是一种求解机器人末端执行器位置对关节角度导数的方法。对n个关节的机器人，*J*的每一列代表相应的关节速度对于末端线速度和角速度的传递比，由文献[2]中的微分变换法，我们得到每一列的表达式如Figure 2。

$$J_{li}=\begin{bmatrix}(\boldsymbol{p}\times\boldsymbol{n})_z\\(\boldsymbol{p}\times\boldsymbol{o})_z\\(\boldsymbol{p}\times\boldsymbol{a})_z\end{bmatrix}(\text{转动关节}i),J_{li}=\begin{bmatrix}n_z\\o_z\\a_z\end{bmatrix}(\text{移动关节}i)$$

$$J_{mi}=\begin{bmatrix}n_z\\o_z\\a_z\end{bmatrix}(\text{转动关节}i)\qquad,J_{mi}=\begin{bmatrix}0\\0\\0\end{bmatrix}(\text{移动关节}i)$$

Fig. 2 微分变换法

在实践中，对${}^0T_{end}$中的XYZ项分别对关节各变量求导，得到一个3*1的J_v向量共4个，代表末端线速度与该关节速度传递比。

从关节n到关节n+1的矩阵中提取${}^nT_{n+1}(3,1:3)$得到3*1的J_ω向量共4个，代表末端角速度与该关节速度传递比。

合并得到一个6*4的矩阵，在灵巧手的逆运动学求解中，手指末端的姿态并不需要考虑，则*J*可以进一步缩减为由J_v构成的3*4矩阵。运用MATLAB 的符号运算实现。用*diff(T0end, theta)* 求导，用*simplify(J_matrix)* 简化。最后我们得到*J*矩阵:

$$\begin{pmatrix}30\sin(\theta_1-\theta_2)\,\sigma_3 & -30\cos(\theta_1-\theta_2)\,\sigma_2 & -30\cos(\theta_1-\theta_2)\,\sigma_4 & -30\cos(\theta_1-\theta_2)\,\sigma_6\\ -30\cos(\theta_1-\theta_2)\,\sigma_3 & -30\sin(\theta_1-\theta_2)\,\sigma_2 & -30\sin(\theta_1-\theta_2)\,\sigma_4 & -30\sin(\theta_1-\theta_2)\,\sigma_6\\ 0 & -30\,\sigma_5-\sigma_1-30\cos(\theta_3) & -30\,\sigma_5-\sigma_1 & -30\,\sigma_5\end{pmatrix}$$

where

$$\begin{aligned}\sigma_1&=30\cos(\theta_3+\theta_4)\\ \sigma_2&=\sigma_6+\sin(\theta_3+\theta_4)+\sin(\theta_3)\\ \sigma_3&=\sigma_5+\cos(\theta_3+\theta_4)+\cos(\theta_3)\\ \sigma_4&=\sigma_6+\sin(\theta_3+\theta_4)\\ \sigma_5&=\cos(\theta_3+\theta_4+\theta_5)\\ \sigma_6&=\sin(\theta_3+\theta_4+\theta_5)\end{aligned}\tag{9}$$

2.2.2 求解数值解

将求解逆解分装成函数IK_Sol供三支手指调用。逻辑如下：

- 获取机械臂末端当前位置XYZ
- 计算位置误差*p_err*
- 判断误差$Loss = norm(p_err)$是否在可接受范围内，如果是则结束迭代
- 计算雅可比矩阵*J*并计算角度修正量$dth = learning_rate * pinv(J) * p_err$
- 更新关节角度$q = q + dth$并应用关节空间约束
- 返回新的关节角度*q*和是否完成标志*done*

做出以下工作优化解算速度,单次迭代时间为$4.3*10^{-4}$秒。单根手指实现2325hz的迭代速度。

- 使用一个明确的雅可比矩阵而不通过叉乘计算
- 通过匿名函数*matlabFunction(J_matrix)*;绕过符号代入*subs()*需要调用的函数库
- 去除重复更新Link中的*A*矩阵；减少重复画图
- 通过探查器生成火焰图，优化耗时函数
- 通过*MATLAB Coder*插件将*Matrix_dh*转换成脚本函数

3 抓取规划

3.1 工作空间绘制

显然，目标必须在工作空间中才能被抓取，这也意味着我们需要测定我们的工作空间。通过蒙特卡洛法绘制了灵巧手的工作空间如Figure 3。可以看到在该构型下，单手指的工作空间呈现出二分之一个碗的形状。进一步的，我们通过alphaShape 多

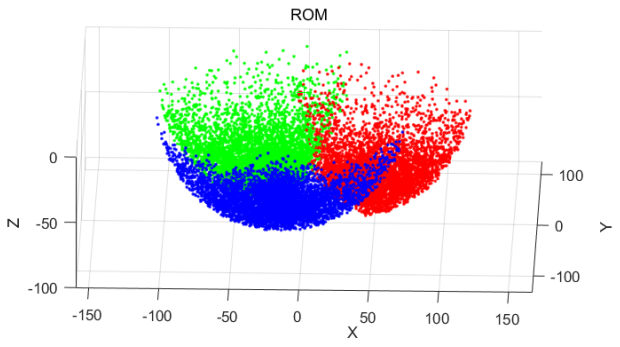

Fig. 3 工作空间-随机点

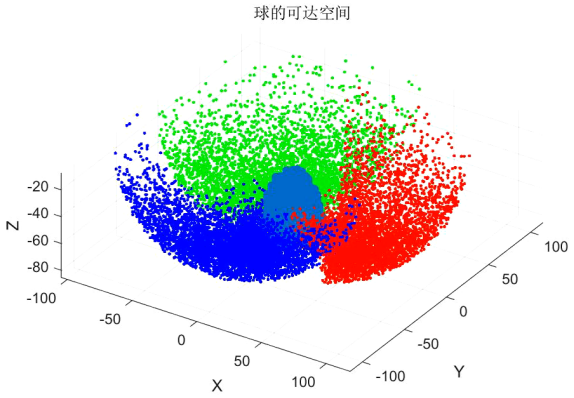

Fig. 5 20mm可抓取范围

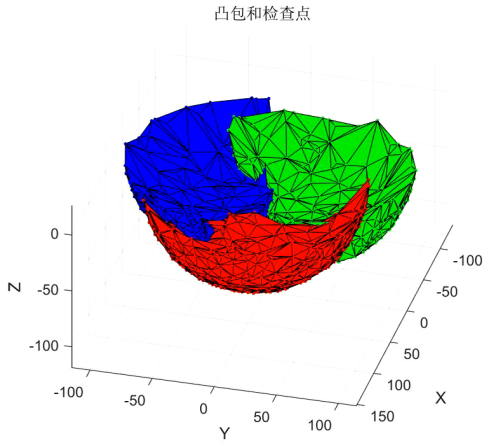

Fig. 4 工作空间-alphaShape多面体

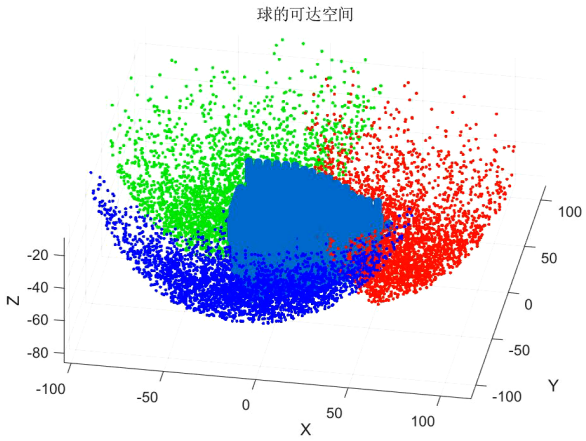

Fig. 6 40mm可抓取范围

面体来建立对工作空间的判断对如Figure 4。具体的判定方法将会在后文提及。

3.2 可抓取空间——以球为例

对于一个待抓取的物体，自然的我们需要在物体上设计我们的抓取点。对于三指灵巧手而言，用三指抓取物体有以下的必要不充分条件：必须有三个点既在物体上又在对于手指的工作空间中。设计了如下的判断标准：$any(pdist2(workspace\_points(), ball\_center) < ball\_radius)$。我们用20mm与40mm的球测试得到Figure 5与Figure 6。蓝色实心块则代表着对应半径的球的球心所在的范围。

3.3 抓取点规划

对于任意物体的抓取点规划总遵循以下的方法：

- 生成候选接触点：在物体表面上生成候选接触点，将候选接触点划分到靠近手指的三个象限
- 检查候选接触点是否在手指的工作空间内：使用*inshape(alphaShape , point)*检查每个象限的候选接触点是否在对应手指的工作空间内
- 判稳：形成稳定的抓取三角形：遍历所有可能的接触点组合,计算形成的三角形形心与物体之间的距离。输出最佳接触点组合

针对球体，我们通过球坐标系来生成下半圆的候选点如Figure 7。针对正方体，我们通过每个面的中间50%部分来生成候选点如Figure 8。具体实现详见附件代码中的*Planning_point.m*。

3.4 抓取力规划

设计了判稳的标准：施加给物体的总力与总力矩接近0，效果如Figure 9。代码详见附件代码中的*Planning_target_torque.m*。这里只设计了末端力

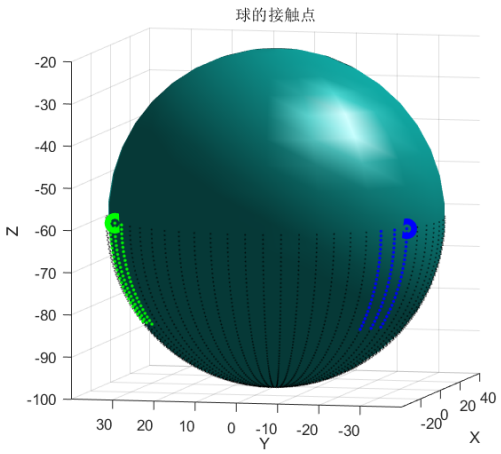

Fig. 7 球的接触点

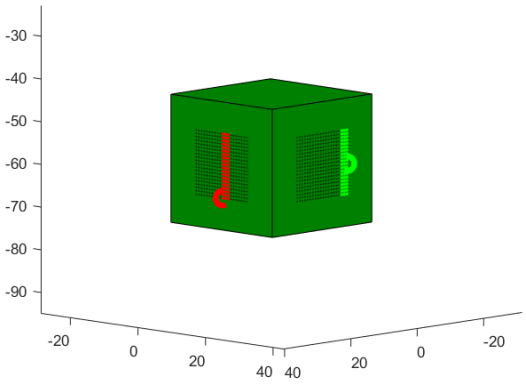

Fig. 8 方块的接触点

与力矩，关节扭矩可以通过以下公式得到：

$$\tau = \mathbf{J}_v^T \mathbf{f} + \mathbf{J}_\omega^T \mathbf{m}$$

其中τ代表关节的扭矩，J_v、J_ω代表雅可比矩阵的速度分量与角速度分量，f代表末端力，m代表末端力矩。

求解末端力与力矩的步骤如下：

- 初始化随机方向向量：生成三个随机的三维向量作为初始猜测值，分别对应三个接触点的力方向。对这些向量进行单位化处理。
- 定义目标函数：使用匿名函数定义目标函数，该目标函数计算物体所受的总力矩和总力。
- 优化：使用*fmincon*函数进行优化，使目标函数最小化，得到最佳的力方向向量。
- 提取优化后的力方向：从优化结果中提取每个接触点的最佳力方向向量。
- 计算总力和总力矩：计算所有接触点的力之和以及所有接触点的力矩之和。
- 检查平衡条件：如果总力矩的范数小于阈值且总力的范数小于阈值，输出力矩与力平衡信息，否则输出力矩或力不平衡信息。

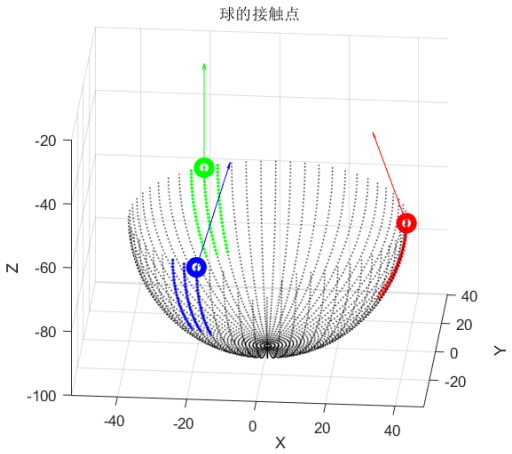

Fig. 9 接触力

3.5 抓取实例

下面的Figure 10与Figure 11演示了实际代码抓取的效果。均实现了对抓取点的规划以及力的规划，并通过逆解迭代求解到达目标点。

4 轨迹规划

4.1 末端位置RRT*算法

*RRT** 是 *RRT* 的改进版本，旨在找到更短的路径，如Figure 12所示。*RRT** 的主要优势是它能够逐步改进路径，最终趋向于找到最优解。它遵循以下的逻辑：

- 初始化：与*RRT* 相同，从起点开始，构建一个树状结构。
- 随机抽样：在搜索空间中随机选择一个点（随机点或目标点）。判断其是否在工作空间内，如不在则重新随机生成。
- 最近邻搜索：找到树中距离随机点最近的节点。
- 路径探索：尝试从最近邻节点向随机点延伸*N*步长，如无碰撞则得到新节点。
- 更新树：找到新节点周围一定半径内的所有节点。从这些节点中选择使得路径总长度最小的节点作为新节点的父节点。

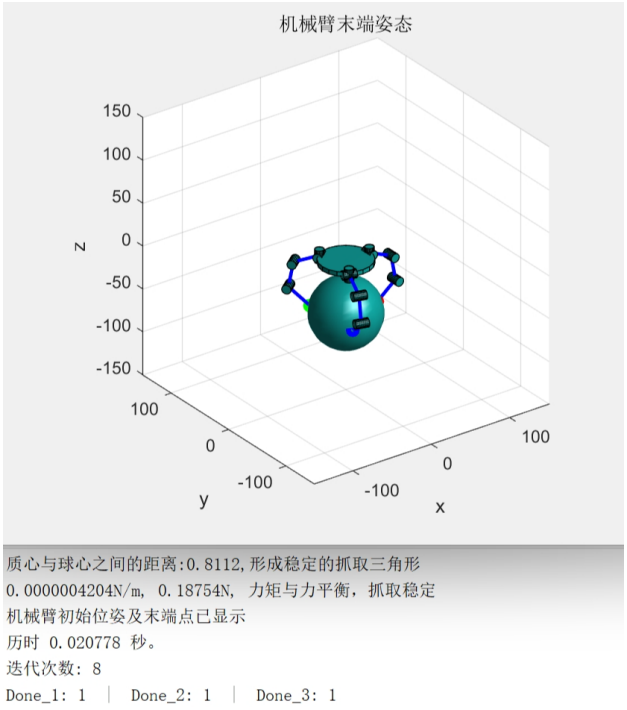

Fig. 10 球的抓取

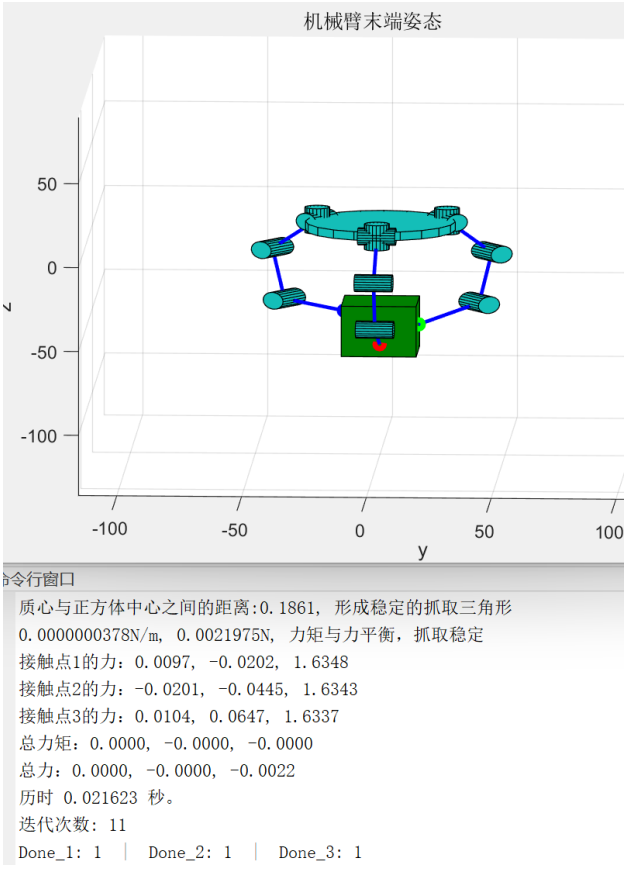

Fig. 11 正方体的抓取

- 迭代：重复上述步骤，直到找到终点或满足停止条件。

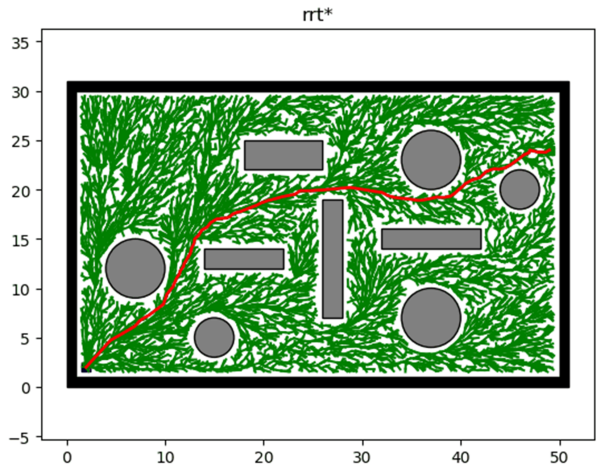

Fig. 12 RRT*算法

4.2 规划实例

下面的Figure 13与Figure 14演示了*RRT*规划与*RRT**规格的效果。均实现了初始点到达抓取点的规划。对比可见，*RRT*相比*RRT**，给出来的路径规划有着更长且更不平滑的效果。在多次实验中也观察到*RRT*给出路径的长度更为随机以及呈现出更不效率的一面。

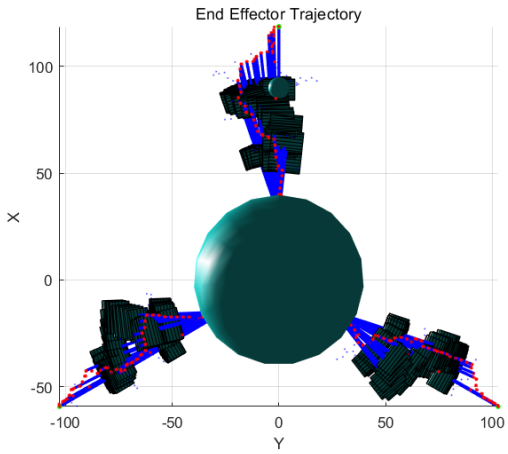

Fig. 13 *RRT*实例

5 机械臂装载实例

使用例程中的*PUMA560*六自由度机械臂进行展示。实现机械臂的正逆运动学(考虑姿态误差)，并实现末端装载灵巧手的抓取。具体实现详见附件代码中的*Catch_with_arm.m*。实现的方法如下：

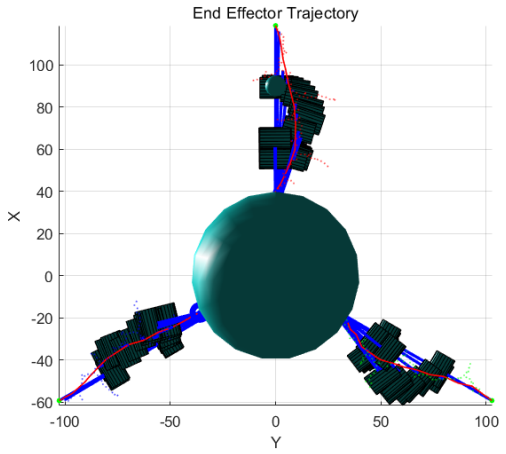

Fig. 14 *RRT**实例

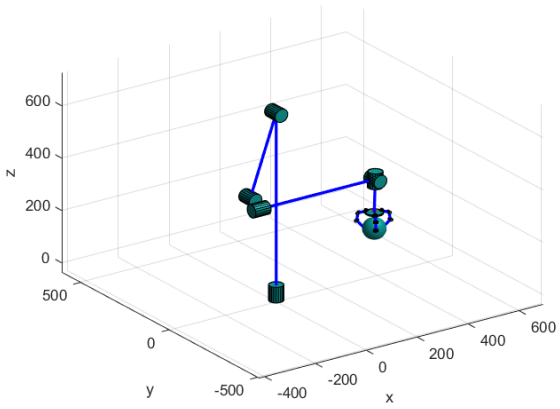

Fig. 15 机械臂装载实例

- 通过机械臂.A*中间变换矩阵*灵巧手基座.A，将灵巧手安装在机械臂的末端。
- 通过*DHfk_J_Puma560()*实现机械臂的正解，通过*Arm_IK_Sol()*实现机械臂的逆解
- 通过*Planning_point()*实现对抓取点的规划
- 通过*Planning_target_torque()*实现对抓取力的规划
- 通过*IK_Sol()*实现对灵巧手的逆解

6 总结与讨论

通过本次课程，我系统地学习到了一个机械机构如何从设计到运行的全过程。感谢老师的悉心指导让我完成了本次课程的工作：设计思路、正运动学、逆运动学、抓取规划、RRT算法以及机械臂装载。

针对于本次工作仍有需要完善的部分，将会在课后继续学习相关内容：

- 4334的运动规划
- RRT的关节避障问题
- 手指的力控
- 对抓取稳定性更完备的证明
- 以施加最大力为代价的抓取姿态控制
- 装载在机械臂下，z轴力空间的变化（重力应由机械臂提供补偿）

参考文献

[1] Llop-Harillo I ,Antonio Pérez-González, Starke J ,et al.The Anthropomorphic Hand Assessment Protocol (AHAP)[J].Robotics and Autonomous Systems, 2019, 121:103259.DOI:10.1016/j.robot.2019.103259.

[2] 蔡自兴.机器人学基础.第2版[M].机械工业出版社,2015.

[3] 梶田秀司,管贻生.仿人机器人[M].清华大学出版社,2007.

[4] JohnJ.Craig,等.机器人学导论[M].机械工业出版社,2006.

[5] HAN M S, HARNETT C K. Journey from human hands to robot hands: Biological inspiration of anthropomorphic robotic manipulators[J/OL]. Bioinspiration Biomimetics, 2024, 19(2): 021001. DOI:10.1088/1748-3190/ad262c.

[6] CHEN W, WANG Y, XIAO Z, et al. Design and experiments of a three-fingered dexterous hand based on biomechanical characteristics of human hand synergies[J]. 2022, 27(5).