

**毕业设计（论文）**

|  |  |
| --- | --- |
| 题 目： | 可编辑调试的机械臂数字孪生可视化系统构建 |
| 学 院： | 航空制造工程学院 |
| 专业名称： | 智能制造工程 |
| 班级学号： | 20037207 |
| 学生姓名： | 梁经伟 |
| 指导教师： | 汪佳良、黄华 |

2O24年5月

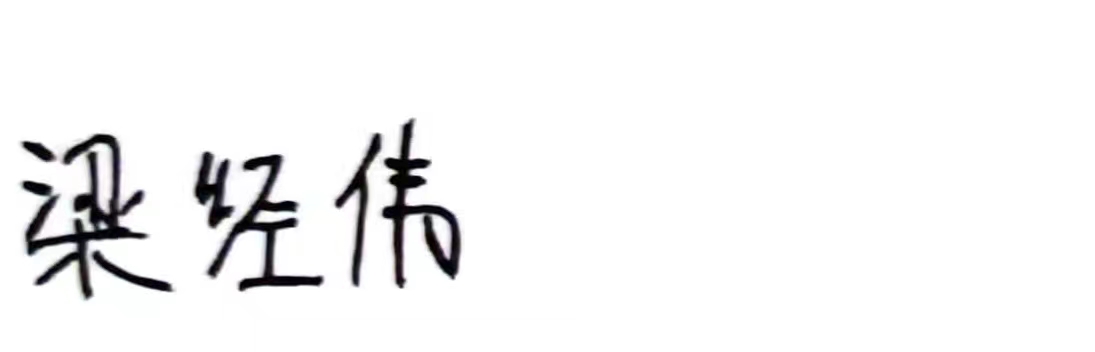
****

## 毕业设计（论文）任务书

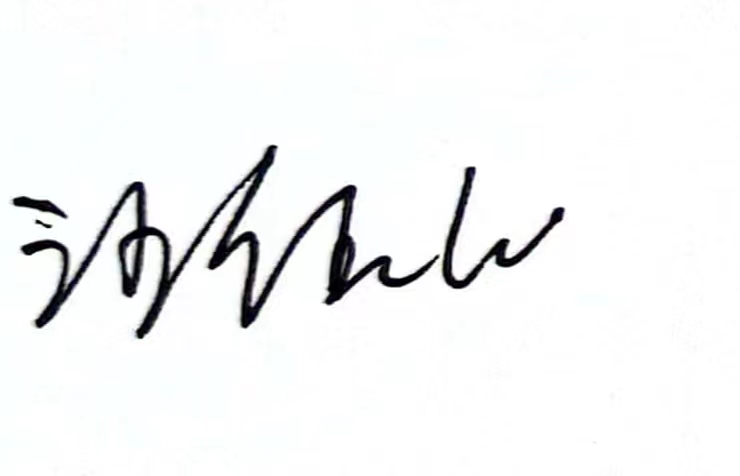
|  |
| --- |
| **I、毕业设计(论文)题目：** |
| **可编辑调试的机械臂数字孪生可视化系统构建** |
| **II、毕 业设计(论文)使用的原始资料(数据)及设计技术要求：** |
| 原始资料(数据)包括： |
| 1.机械臂的技术规格：机械臂的尺寸、速度、旋转精度； |
| 2.机械臂的运动学和动力学模型：用于描述机械臂各关节之间的运动关系。 |
| 3.控制系统的设计要求：包括控制算法的选择、反馈机制的设计、控制参数的设定等。 |
| 4.传感器数据：包括位置、速度等传感器采集的实时数据。 |
| 5.软件资源：包括操作系统、开发工具、仿真软件、编程语言等。 |
| 设计技术要求： |
| 1.系统的可编辑性：确保用户能够方便地修改机械臂的模型参数和控制策略。 |
| 2.系统的实时性：仿真模型应能够实时反映机械臂的实际状态，并对控制命令做出快速响应。 |
| 3.系统的准确性：数字孪生模型需尽可能精确地复现机械臂的物理特性和动态行为。 |
| 4.系统的稳定性：确保在长时间运行和各种工况下，系统都能稳定可靠地工作。 |
| 5.用户交互性：设计直观的用户界面，方便用户监控和调整机械臂的运行状态。 |
| **III、毕 业设计(论文)工作内容及完成时间：** |
| 1. 文献综述与需求分析 2周 |
| 2. 系统方案设计 3周 |
| 3.熟悉机械臂模型建立与仿真 1周 |
| 4.熟悉控制系统开发 1周 |
| 5. 系统集成与测试 3周 |
| 6. 数据采集与分析 2周 |
| 7. 系统优化与文档撰写 2周 |

|  |
| --- |
| 8. 答辩准备 1周 |
| **Ⅳ 、主 要参考资料：** |
| [1]陈玉娇,曾诗雨,和红杰等.工业机器人码垛数字孪生系统的研究与实现[J].计算 |
| 机集成制造系统,2023,29(06):1930-1940. |
| [2]陈述,鲁世立,王建平等.融合Unity3D的缆索起重机安全运行数字孪生模型构建方 |
| 法[J].中国安全生产科学技术,2024,20(01):154-159. |
| [3]周高伟,沙杰,刘梦园等.基于数字孪生的加工生产线虚实交互技术研究[J/OL].机 |
| 电工程:1-9[2024-03-08]. |
| [4]杨思楠. 机械臂3D陶土打印技术研究及其数字孪生工厂构建[D].石家庄铁道大 |
| 学,2023. |
| [5]李颖,熊冰,桑笑晗,等.沉浸式力学实验数字孪生系统架构研究[J/OL].工程力 |
| 学:1-13[2024-04-22] |

航空制造工程  **学院** 智能制造工程 **专业 200372 班**

**学生（签名）：** 

**日期： 自 2023年 12** **月 1 8 日 至**  **2024**  **年 5** **月 31 日**

**指导教师（签名）：**

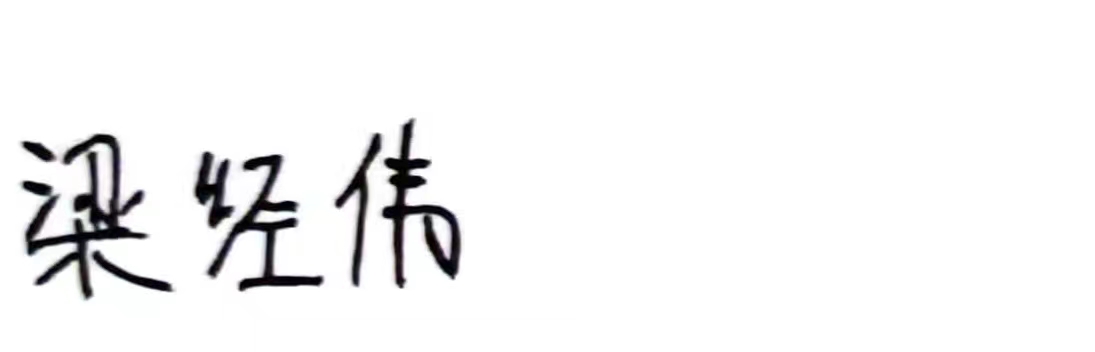
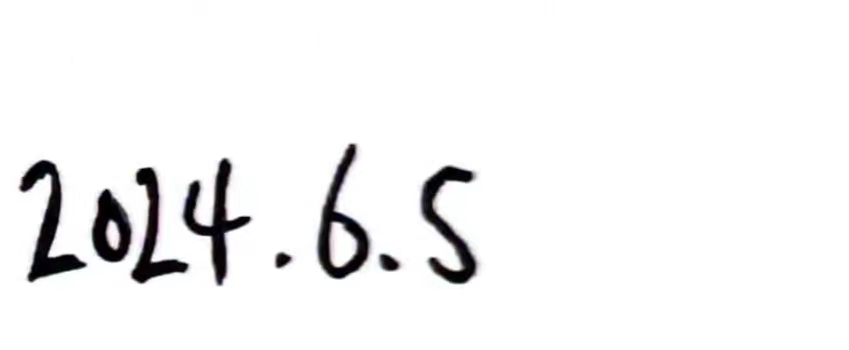
**助理指导教师(并指出所负责的部分)：**

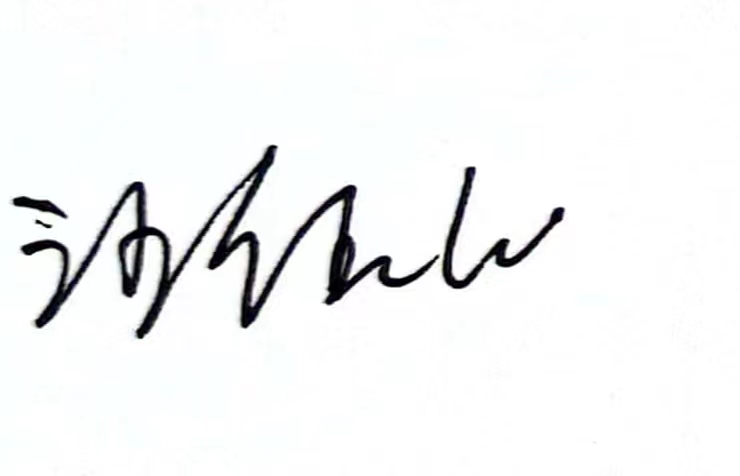
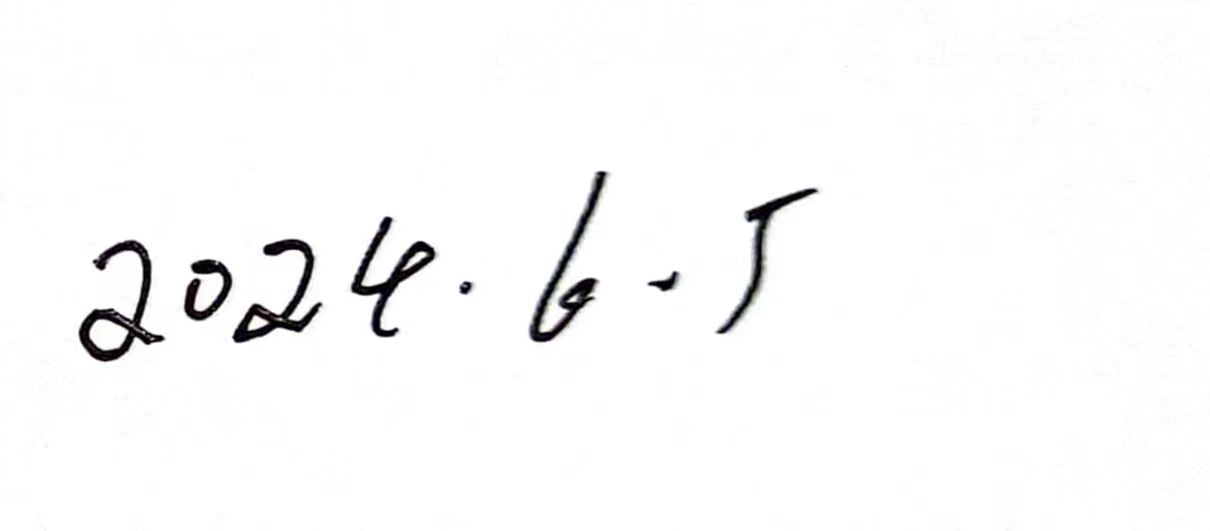
**智能制造工程 系（室） 主任（签名）：**



## 学士学位论文原创性声明

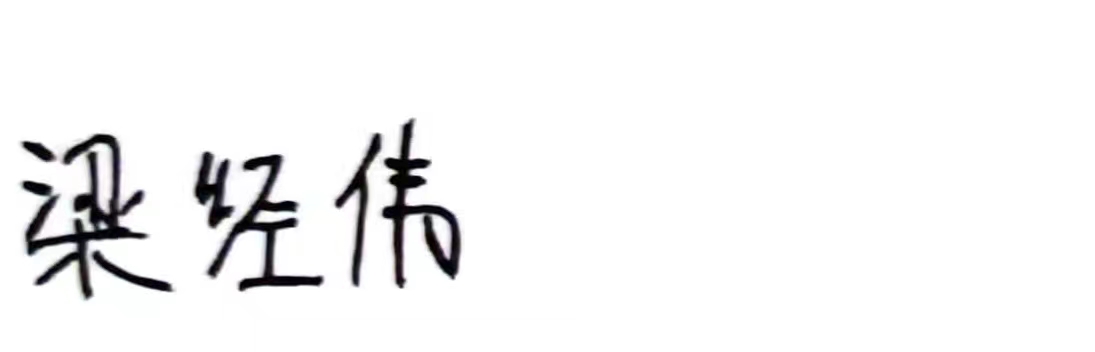
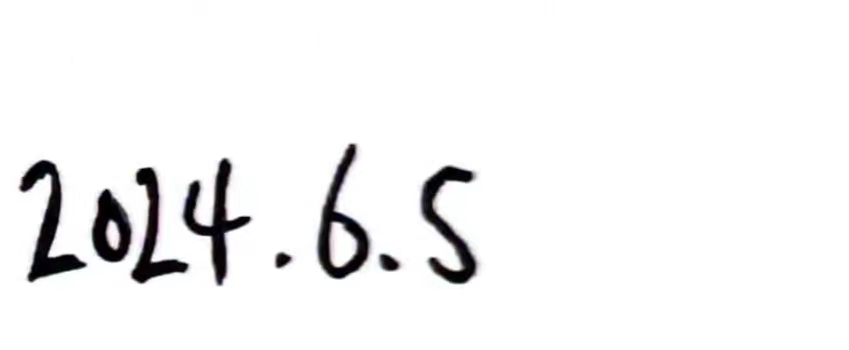
本人声明，所呈交的论文是本人在导师的指导下独立完成的研究成果。除了文中特别加以标注引用的内容外，本论文不包含法律意义上已属于他人的任何形式的研究成果,也不包含本人已用于其他学位申请的论文或成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式表明。坚决杜绝论文买卖、代写、抄袭、剽窃等不良现象，确保毕业设计（论文）质量。本人完全意识到本声明的法律后果由本人承担。

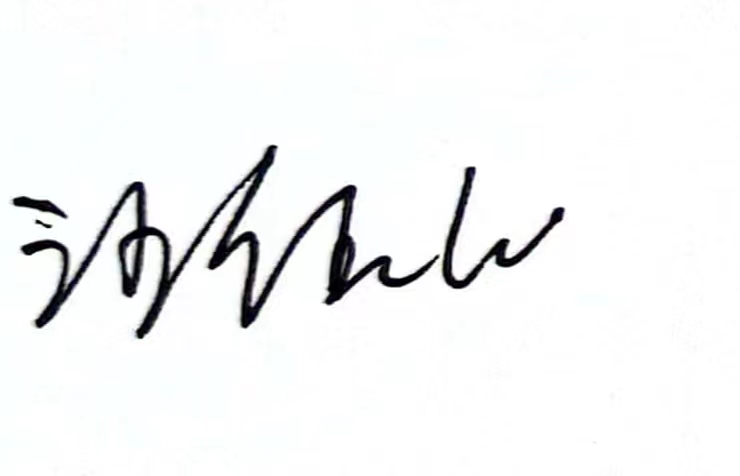
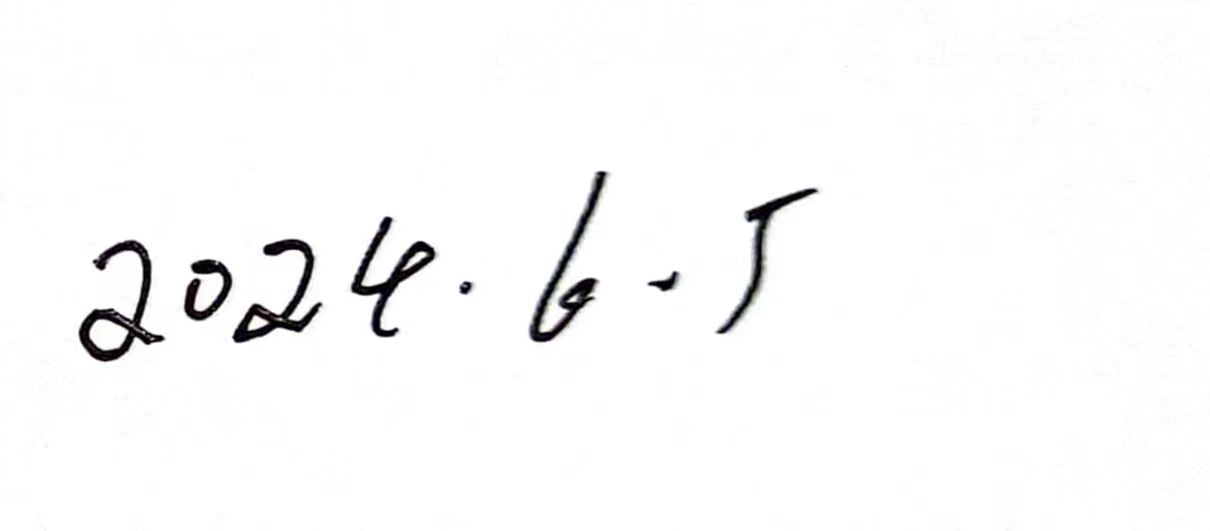
作者签名： 日期：

导师签名：**** 日期：

**学位论文版权使用授权书**

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权南昌航空大学可以将本论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

作者签名： 日期：

导师签名：**** 日期：

**可编辑调试的机械臂数字孪生可视化系统构建**

学生姓名：梁经伟 班级：200372

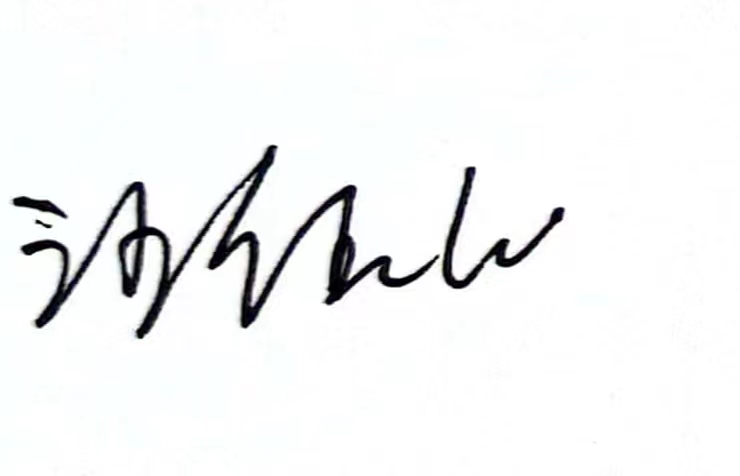
指导老师：汪佳良、黄华

**摘要：**本研究旨在构建一个可编辑调试的机械臂可视化数字孪生系统，以提升机械臂的可视化与可编辑性、实现远程调试与优化以及推动技术创新与发展。

研究采用了Django、Three.js、Redis和WebSocket技术构建数字孪生可视化系统。Django作为后端框架，处理数据管理和分发，提供了稳定可靠的数据处理和业务逻辑支持；Three.js用于前端机械臂数字孪生模型的3D可视化展示；Redis数据库确保了用户注册登录信息的快速存储与访问；WebSocket通讯技术实现了数据的实时传输和同步。系统允许用户通过Django、Three.js、WebSocket和Redis技术实时同步机械臂数据，进行编辑和调试。研究还涉及了物理实体采样、虚拟模型建立、数据交互设计和可视化展示等关键技术。

系统实现了机械臂的远程调试和优化，提升了操作的直观性。通过Three.js，用户可以在网页上实时调整机械臂动作和参数。利用Redis和WebSocket技术，实现了高效的数据读写和通讯，确保了系统的实时性和交互性。系统通过改进的粒子群算法优化了机械臂的运动路径，以最小化能量消耗，实现了机械臂的精确控制。

**关键词：**数字孪生系统；机械臂；可编辑调试；可视化；

**指导老师签名：**

**Editable and Debuggable Robotic Arm Visual Digital Twin System Construction**

Student name : Liang Jingwei Class: 200372

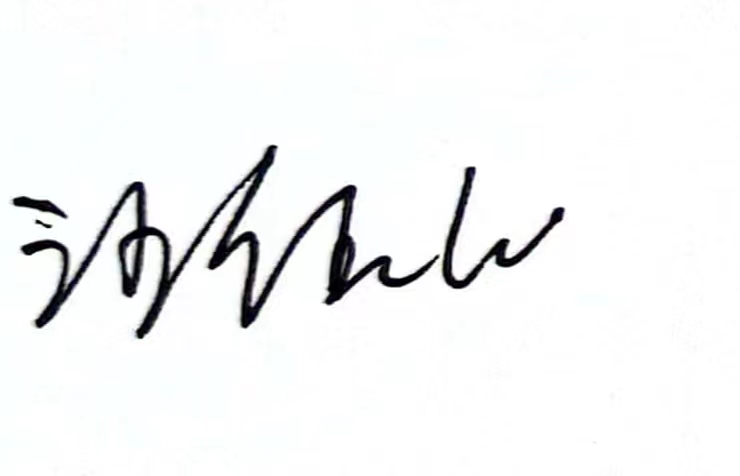
Supervisor: Wang Jialiang、Huang Hua

**Abstract:** The purpose of this study is to construct a visual digital twin system of robotic arm that can be edited and debugged, so as to improve the visualization and editability of robotic arm, realize remote commissioning and optimization, and promote technological innovation and development.

Django, Three.js, Redis and WebSocket technologies were used to build a digital twin visualization system. As a back-end framework, Django handles data management and distribution, providing stable and reliable data processing and business logic support. Three.js 3D visualization of the digital twin model of the front-end robotic arm, Redis database ensures the fast storage and access of user registration and login information, and WebSocket communication technology realizes real-time data transmission and synchronization. The system allows users to synchronize robot data in real time through Django, Three.js, WebSocket, and Redis technologies for editing and debugging. The research also involves key technologies such as physical entity sampling, virtual model establishment, data interaction design and visual display.

The system realizes the remote debugging and optimization of the robot arm, which improves the intuitiveness of the operation. Through Three.js, users can adjust the action and parameters of the robot arm in real time on the web page. Redis and WebSocket technology are used to realize efficient data reading and writing and communication, which ensures the real-time and interactivity of the system. The motion path of the manipulator is optimized by improved particle swarm optimization algorithm to minimize the energy consumption, and the precise control of the manipulator is realized.

**Keywords:** Digital twin system；Robotic arm；Editable debugging；Visualization

**Signature of Supervisor：**

目 录

[第1章 绪论 1](#_Toc20608)

[1.1研究背景与意义 1](#_Toc13435)

[1.2 国内外研究现状 1](#_Toc9183)

[1.3研究内容与方法 3](#_Toc7398)

[1.3.1数字孪生框架搭建 4](#_Toc20652)

[1.3.2物理实体采样 4](#_Toc15230)

[1.3.3虚拟模型建立 5](#_Toc136)

[1.3.4数据交互设计 5](#_Toc6424)

[1.3.5可视化展示 5](#_Toc28620)

[第2章 前端技术 6](#_Toc17464)

[2.1 前端技术框架 6](#_Toc20557)

[2.1.1 Three.js 6](#_Toc5940)

[2.1.2 Unity 3D 7](#_Toc11127)

[2.1.3 Vue 3D 7](#_Toc29813)

[2.2 数字孪生前端系统 7](#_Toc1559)

[2.2.1 UI设计和three.js集成 8](#_Toc20878)

[2.2.2 编辑功能 11](#_Toc23427)

[2.2.3 WebSocket通信 14](#_Toc28742)

[2.3本章小结 15](#_Toc29449)

[第3章 后端技术 17](#_Toc23757)

[3.1 后端技术框架 17](#_Toc27080)

[3.1.1 Django 17](#_Toc17071)

[3.1.2 Node.js 19](#_Toc23651)

[3.1.3 Flask 19](#_Toc30273)

[3.2数字孪生后端系统 20](#_Toc14119)

[3.2.1 Django后端框架搭建 21](#_Toc16032)

[3.2.2 WebSocket服务搭建 23](#_Toc10406)

[3.2.3 数据处理和交互 26](#_Toc3191)

[3.3本章小结 29](#_Toc20940)

[第4章 数据库技术 30](#_Toc27686)

[4.1用户注册登录 31](#_Toc26617)

[4.2本章小结 34](#_Toc13834)

[第5章 优化算法 36](#_Toc24751)

[5.1 正向运动学分析 36](#_Toc21987)

[5.2 逆向运动学分析 37](#_Toc15307)

[5.3 改进粒子群算法 37](#_Toc2747)

[5.4 应运算法 39](#_Toc26721)

[5.5本章小结 47](#_Toc19132)

[第6章 总结与展望 48](#_Toc25018)

[参考文献 51](#_Toc4804)

[致 谢 53](#_Toc19717)

# 绪论

## 1.1研究背景与意义

数字孪生的概念最初可追溯到 2002 年美国大学的产品全生命周期管理（PLM）课程中，由迈克尔·格里夫斯博士提出的“镜像空间”（Mirrored Space）的理念。他于2003年发表了一篇名为《数字孪生：通过虚拟工厂实现制造卓越》的论文。在《Y Replication》的讲话中进一步探讨了这一概念。随着时间的推移，科技的飞速发展为数字孪生注入了强大的动力，使其在各个领域熠熠生辉。其发展历程如图1. 1。

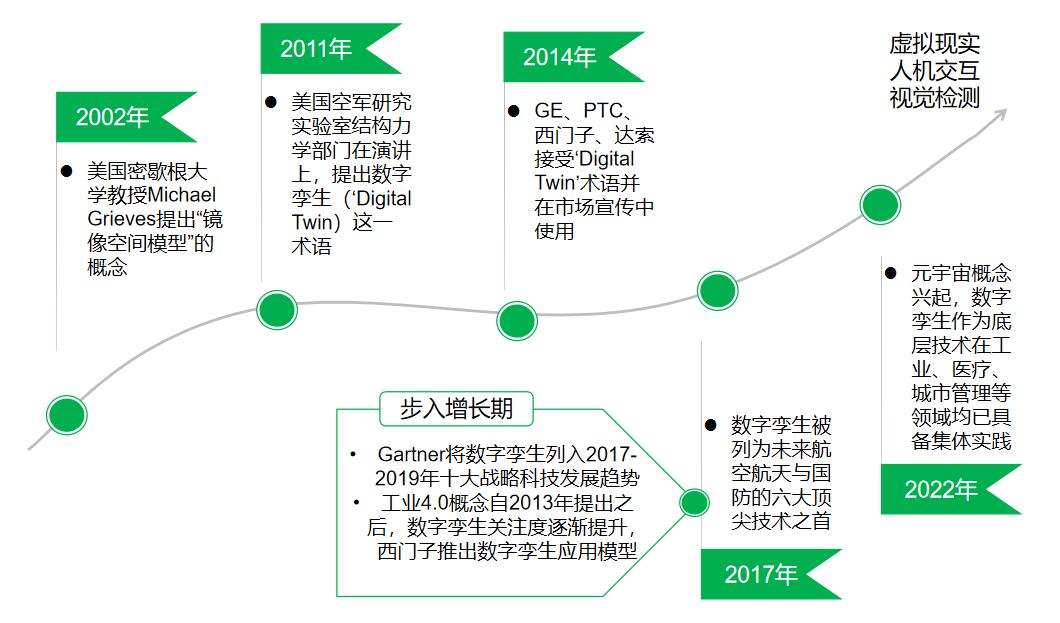


图1. 1 数字孪生发展历程

在智能制造、智能建造、智慧医疗、智慧城市等众多领域，数字孪生技术都展现出了其独特的魅力和价值。它像是一位无所不能的魔法师，为智能运维、虚拟调试、异常诊断、风险预测、决策辅助、系统优化等方面带来了前所未有的便利和突破。数字孪生技术已经成为推动企业数字化转型、提升生产效率、促进数字经济发展不可或缺的重要工具。

## 1.2 国内外研究现状

在经济社会的数字化转型进程中，数字孪生的重要性逐渐凸显，尤其在工业4.0时代的背景下，它成为推动社会进步的重要力量。中国政府高度重视数字孪生技术，出台了《中国数字孪生城市行业研究》、《数字孪生城市白皮书》以及《十四五》规划等一系列政策文件，明确了数字孪生在城市建设和制造业中的核心地位。其中，《数字孪生城市白皮书》更是指出，《十四五》规划纲要提出“探索建设数字孪生城市”的宏伟蓝图，并要求在后续的政策实施中不断提升其整体性、系统性和协同性。

数字孪生技术已成为制造业不可或缺的一环，它能即时模拟物理设备的实时状态，极大地促进了设备的维护、故障预警和寿命管理。随着AI和机器学习技术的迅速进步，数字孪生的数据收集和分析能力有了显著增强，能更细致地了解如机械臂等设备的性能特点。这种提升有助于精准预测并避免潜在故障，减少维修成本，也有助于提升生产效率，为企业带来更大的经济效益。从2022年开始，中国的数字孪生市场越来越受欢迎，市场规模迅速增长，达到了104亿元，而且增长速度很快，增加了35%。预测显示，到2024年，这一市场的规模会大幅增加到237亿元。在产业链中，一些顶尖的公司如海康威视和华为，凭借其高超技术成为了行业的佼佼者。而在应用方面，微软和达索等公司则通过他们创新的方案推动了整个行业的发展。目前，大多数相关企业主要集中在北京、江苏这些经济繁荣的地方，形成了一个产业集聚的良好局面。具体数据如表1. 1。

| 项目 | 中国 | 国外 |
| --- | --- | --- |
| 数字孪生市场规模（2022年） | 104亿元 | 165亿美元 |
| 数字孪生市场规模同比增长率（2022年） | 35.0% | 33.50% |
| 数字孪生市场规模预测（2024年） | 237亿元 |  |
| 产业链基础支撑层高端参与者 | 海康威视、华为、寒武纪等 | 高通、英特尔、博世、意法半导体等 |
| 仿真分析和模型构建层主要软件市场 | 外资产品占据 | ANSYS、海克斯康、Altair、西门子等 |
| 应用层代表企业 | 微软、达索、ANSYS、上海优也等 |  |
| 数字孪生注册企业主要分布省市 | 北京、江苏等为首的发达省市 |  |
| 数字孪生企业数量（北京市） | 370家以上 |  |
| 代表性企业分布情况 | 北京、浙江、广东等地代表性企业较多 | 美国专利技术储备靠前，研究集中在军民两大领域 |
| 数字孪生技术应用的领域 | 工业制造、城市规划、物流运输、医疗健康、能源管理、航空航天、金融等 |  |

表1. 1 2022-2024年国内外数字孪生行业概况的统计表

数字孪生技术的风潮正如狂风巨浪般席卷而来，特别是自2023年以来，随着“元宇宙”概念的如日中天，它更是成为了学术界、工业界等各方瞩目的焦点。这一技术犹如一座神奇的桥梁，成功地将物理世界与数字世界紧密相连，实现了二者之间的虚实交融。

随着时代的进步和科技的不断发展，机械臂在工业生产、医疗服务、航天探索等多个领域的应用日益广泛。传统的机械臂系统往往存在调试难度大、操作复杂等问题，这在一定程度上限制了机械臂的灵活性和应用范围。因此，构建一个可编辑调试的机械臂数字孪生系统成为了当前研究的重要课题。构建可编辑调试的机械臂数字孪生系统的研究目的在于提升机械臂的可视化与可编辑性、实现远程调试与优化、推动技术创新与发展以及培养专业人才等方面。通过本课题的研究，期望能够为机械臂技术的未来发展奠定坚实基础，为工业自动化和智能制造领域的发展做出重要贡献。可以实现简化机械臂的可视化和调整、实现机械臂的远程调试和促进机械臂技术的创新。

## 1.3研究内容与方法

在国内外相关研究中，有诸多文献与本研究课题密切相关。例如：在 Jadon等人[7]的研究中介绍了在Django框架下开发Web应用的方法，该框架可用于构建可编辑调试的机械臂数字孪生系统中的后端部分。其次，Martínez等人[18]在研究中讨论了基于Three.js的设备控制系统的监控屏应用和发展前景，为构建数字孪生体提供了有益信息。此外，黄平等人[9]的研究中研究了基于能量法的空间机械臂姿态控制稳定性，为数字孪生体的稳定性研究提供了借鉴。数字孪生具体框架如图1. 2。

机械臂的智能化和远程调试需求随着技术发展变得更重要。本课题利用Django、Three.js、Redis和WebSocket技术构建了一个可编辑调试的机械臂数字孪生系统，这对机械臂的研究有诸多好处，如：通过Three.js使操作更直观；使用Django和WebSocket，可以实现远程控制和监控机械臂等。该研究将Django作为后端框架，利用其强大的数据处理和WebSocke快速通讯的能力，实现对系统数据的管理和分发。借助three.js技术，实现了系统数字孪生模型的可视化展示，使用户可以通过直观的界面监测系统运行状态。还采用了redis作为数据库，通过其快速的数据读写能力，确保了系统数据的实时性和一致性。结合Django和three.js等技术实现可调式机械系统的数字孪生和后端数据同步是一种有效的方法，能够为工程实践提供重要的参考价值。

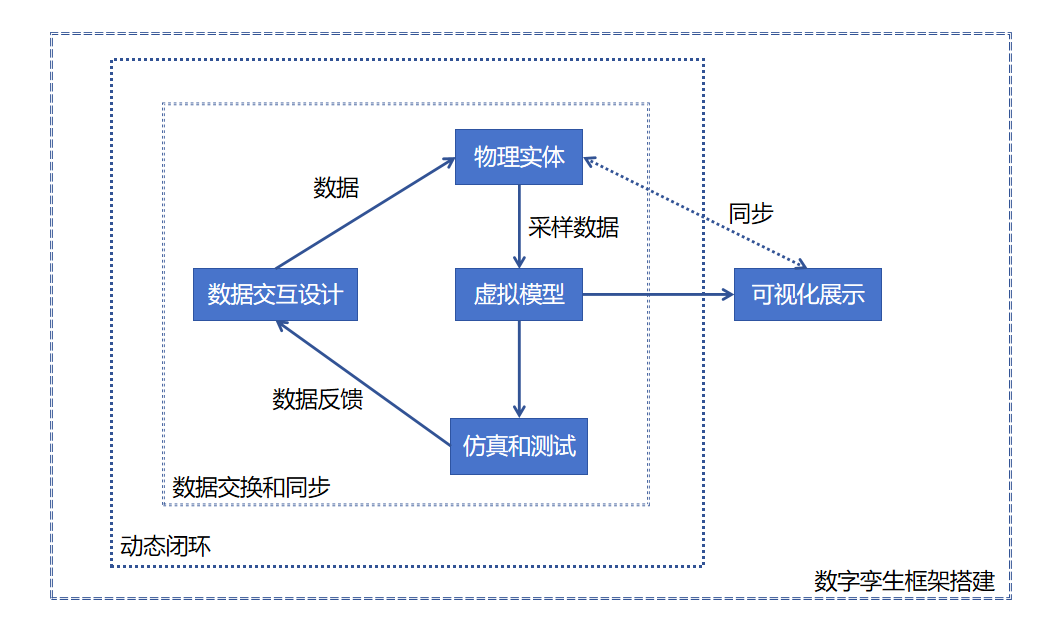


图1. 2 数字孪生框架图

### 1.3.1数字孪生框架搭建

在构建可编辑调试的机械臂数字孪生系统中，数字孪生框架的搭建是基础。基础框架通常由多层架构组成，包括物理实体层、数据层、服务层以及孪生层。每一层都承载着不同的功能和责任，确保数字孪生系统能够有效地映射和模拟物理机械臂的行为和状态。例如，韩冬阳等人[13]提出了正反向数字孪生监测控制框架，该框架通过信息物理系统（CPS）整合物理真实数据与孪生虚拟数据，实现生产过程的全闭环管控；陈述等人[3]通过融合Unity3D构建了缆索起重机的数字孪生模型，该模型不仅包含了虚拟场景的搭建，还实现了与物理实体的实时数据交互，为数字孪生框架提供了三维可视化的能力。

### 1.3.2物理实体采样

物理实体采样涉及对机械臂及其作业环境的实时数据采集，这些数据包括但不限于位置、速度、力矩等关键参数。在周高伟等人[4]的研究中，通过传感器实时采集加工生产线中的数据，并将这些数据传输到数字孪生体中，完成虚实映射。这些数据为数字孪生模型提供了实时更新的依据，确保了模型与物理实体的同步性。韩冬阳等人[13]提到了通过传感器实时采集数据，这些数据作为数字孪生模型的输入，是物理实体采样的重要组成部分。陈述等人[3]通过数字孪生技术获取实时数据，并利用这些数据驱动孪生模型进行模拟仿真，这些实时数据的采集和应用体现了物理实体采样的重要性。

### 1.3.3虚拟模型建立

虚拟模型的建立是数字孪生系统的核心部分，它需要精确地反映物理机械臂的几何结构、物理特性和行为特征。在林润泽等人[6]的研究中，通过三维建模软件和有限元分析，创建了高保真的机械臂虚拟模型，并将其实例化到数字孪生系统中。这些模型不仅包括了机械臂的几何形状，还集成了行为和上下文模型，使得数字孪生能够模拟机械臂的真实行为。陈述等人[3]融合了Unity3D的数字孪生模型构建方法，通过高清渲染管线对模型进行渲染，改善了视觉效果，这为虚拟模型的建立提供了技术支持。

### 1.3.4数据交互设计

数据交互设计确保了物理实体层与数字孪生层之间的信息流动和通信。在韩冬阳等人[13]的研究中，设计了一套OPC 等通讯协议实现数据交互机制，使得数字孪生系统能够接收来自物理实体的实时数据，并根据这些数据进行相应的模拟和控制指令的生成，实现了设备间的互联互通和数据的实时传输。

### 1.3.5可视化展示

可视化展示使得用户直观地观察和分析数字孪生的运行状态和结果。在林润泽等人[6]中，他们利用虚幻引擎（Unreal Engine）构建了机械臂的三维可视化场景，通过实时数据驱动的仿真，用户可以在虚拟环境中监控机械臂的运行情况，包括其位置、姿态和作业状态。这种可视化不仅有助于理解机械臂的当前行为，而且还可以用于预测和优化未来的操作。陈述等人[3]通过高清渲染管线对数字孪生模型进行渲染，改善了视觉效果，提供了直观的可视化展示。

# 第2章 前端技术

数字孪生系统的前端技术意义深远，不仅是数字孪生系统与用户之间沟通的桥梁，更是推动数字化转型和智能化发展的重要力量。

前端技术为数字孪生系统提供了直观、生动的展示界面，使得用户能够轻松地理解和操作这个镜像的复制品。通过先进的可视化技术，数字孪生系统能够镜像出现实世界中的物理实体及其运行状态，帮助用户更全面地了解实体的情况。也提供了丰富的交互功能，使得用户能够与虚拟世界进行实时的互动。这种交互性不仅增强了用户的参与感和体验感，还使得用户能够更加方便地对虚拟世界进行操控和优化，进一步提升了数字孪生系统的实用价值。

前端技术能够实现数据的实时采集和处理，为数字孪生系统提供源源不断的数据支持。通过采集物理实体的实时数据，能够将这些数据进行清洗、整合和格式化，使得数据更加符合数字孪生系统的需求，从而提高了系统的准确性和可靠性。不仅提高系统的可用性和易用性，还推动数字化转型和智能化发展。

## 2.1 前端技术框架

### 2.1.1 Three.js

Three.js 是一个功能强大的 JavaScript 3D 库，它基于 WebGL 技术，旨在简化在 Web 上创建交互式 3D 可视化的过程。由于 WebGL 的底层特性复杂，直接使用 WebGL 编程的门槛较高，Three.js 提供了一个高级的抽象层，使开发者能够更轻松地创建复杂的数字孪生场景和交互式 3D 可视化。其框架图如图2. 1所示

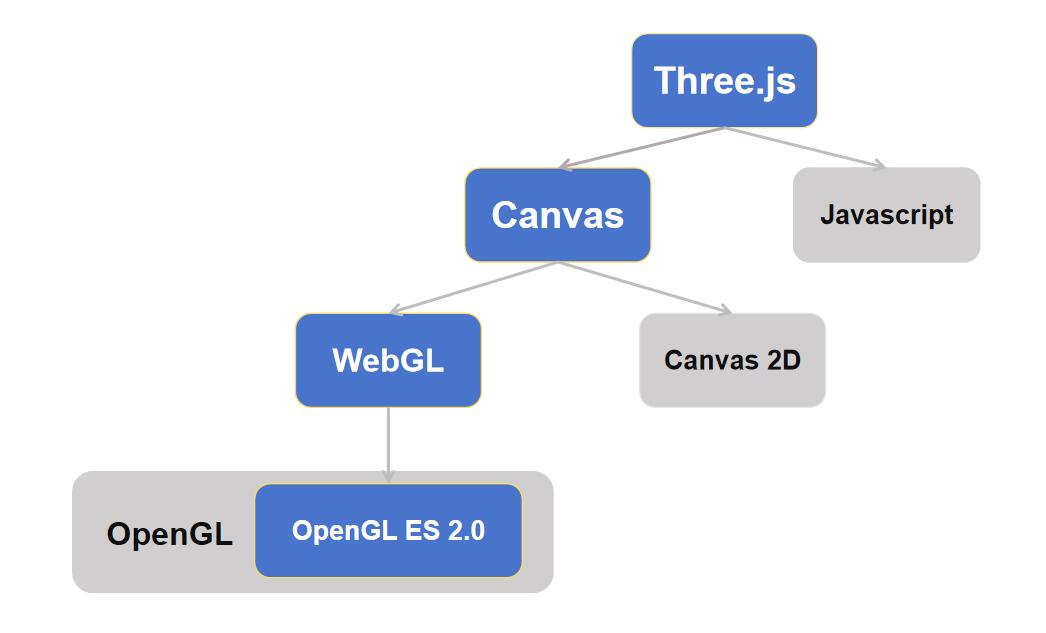


图2. 1 Three.JS技术框架

为开发者提供了丰富的功能和工具，使他们能够在 Web 上创建各种类型的数字孪生场景和交互式 3D 可视化，并且可以通过简单的 API 调用来实现复杂的效果。Martínez等人[18]在研究中讨论了基于Three.js的设备控制系统的监控屏应用和发展前景，为构建数字孪生体提供了有益信息。

### 2.1.2 Unity 3D

Unity 3D是一款跨平台的游戏引擎，具有强大的功能和易用性。除了游戏开发，Unity 3D还广泛应用于创建数字孪生应用的交互式3D场景，为用户提供沉浸式体验和高度定制的解决方案。

提供了丰富的工具和功能，使开发者能够创建高度定制和复杂交互的数字孪生场景，使用户沉浸和真实的体验数字孪生系统。陈述等人[3]融合了Unity3D的数字孪生模型构建方法，通过高清渲染管线对模型进行渲染，改善了视觉效果，这为虚拟模型的建立提供了技术支持。

### 2.1.3 Vue 3D

Vue 3D是一款基于Vue.js的3D库，提供在Vue.js中创建数字孪生应用的便捷方式。借助Vue 3D，利用Vue.js的组件化和状态管理能力，可以轻松地构建复杂的3D场景和交互式应用。其功能强大、易于使用的基于Vue.js的3D库，提供了一种方便快捷的方式来创建数字孪生应用。借助 Vue.js 的组件化和状态管理能力，结合 WebGL、WebVR 等 Web 技术，Vue 3D 能够满足开发者对于复杂 3D 场景和交互式应用的需求，并且易于学习和使用。

## 2.2 数字孪生前端系统

Three.js作为一款轻量级且功能强大的JavaScript库，为开发者提供了丰富的3D图形API，利用Three.js这一强大的3D图形库，创建逼真的机械臂模型，并通过精细的渲染，将机械臂的运动状态实时展示在网页上。前端还负责处理用户的输入指令，将用户的编辑和调试操作实时反映到数字孪生系统上，从而实现了机械臂的远程控制和调试。同时，Three.js与WebGL紧密结合，能够充分利用GPU加速，实现高效的渲染性能。拥有活跃的社区和丰富的文档资源，为开发者提供了强大的技术支持和学习资源。

因此，本课题采用Three.js作为可编辑调试的机械臂数字孪生可视化系统构建的前端。

### 2.2.1 UI设计和three.js集成

**机械臂编辑页面**

编写机械臂编辑页面（playground.html），将Three.js嵌套在可编辑调试的机械臂数字孪生可视化系统构建系统中，编写后默认的机械臂编辑页面如图2. 2所示。

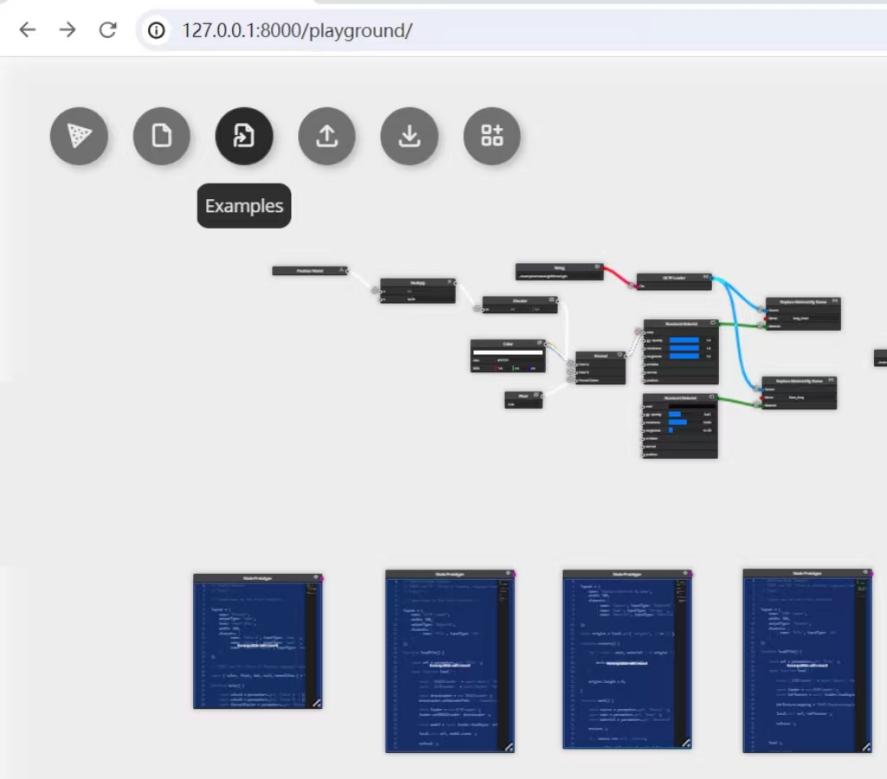


图2. 2 机械臂编辑默认页面

为了方便启动算法以及加载机械臂模型，在playground.html文件中加入算法启动模块以及加载机械臂模块（点击跳转至机械臂模型页面）。

关键代码：

算法启动模块以及加载机械臂模块：

<div class="button-container">

<div>

<button id="algorithmButton">启动算法</button>

</div>

<div>

<button id="loginButton">加载机械臂</button>

</div>

</div>

为了避免手动输入网址http://127.0.0.1:8000/editor/1/（机械臂模型页面），在前端机械臂编辑页面加入了加载机械臂按钮，点击加载机械臂按钮，会自动跳转至机械臂模型页面（http://127.0.0.1:8000/editor/1/）。

关键代码：

const loginButton = document.getElementById('loginButton');

loginButton.addEventListener('click', function() {

window.open('http://127.0.0.1:8000/editor/1/', '\_blank');});

本课题是可编辑调试的机械臂数字孪生可视化系统构建，整个系统具有对机械臂自由度的可编辑性，将2-6轴的josn文件进行预制，放置在playground\examples文件夹中，并且将其添加到机械臂编辑页面的menu中的Examples中，以便于用户选择机械臂的自由度。

关键代码：

改写NodeEditor.js文件

addExamples( 'Robotic arm', [

'6zhou',

'5zhou',

'4zhou',

'3zhou',

'2zhou'

] );

编辑后的机械臂编辑页面（http://127.0.0.1:8000/playground/）如图2. 3所示

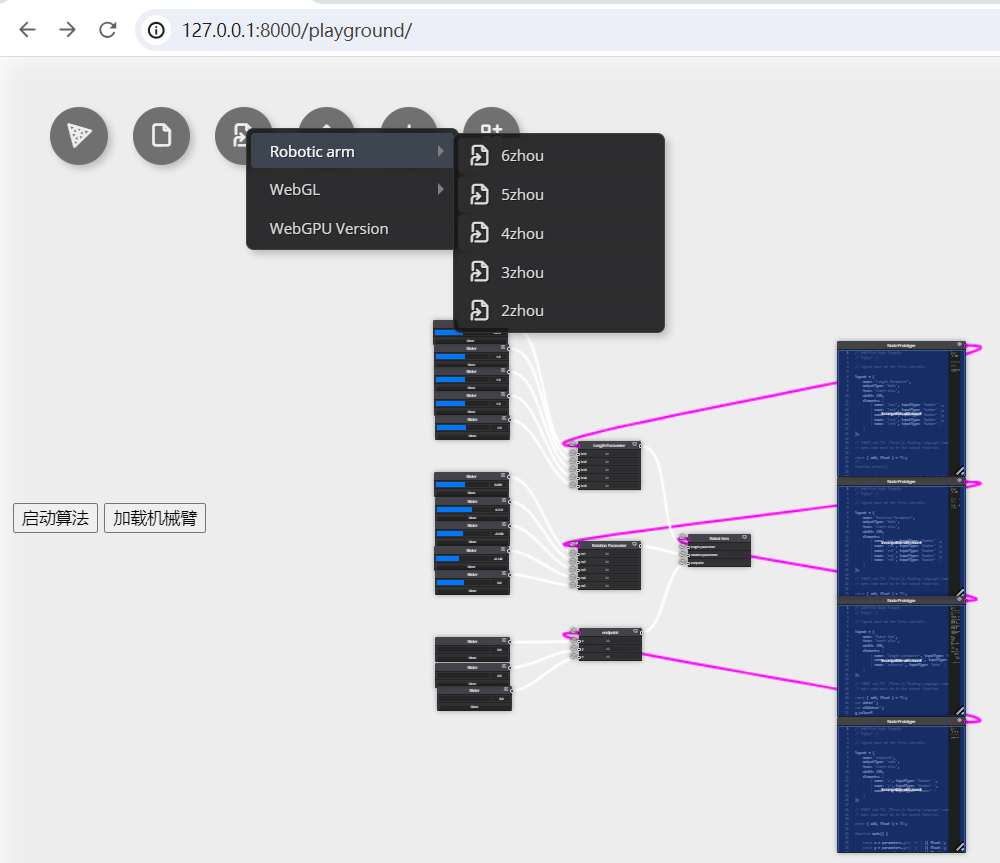


图2. 3 机械臂编辑页面

**机械臂模型页面**

将Three.js嵌套在可编辑调试的机械臂数字孪生可视化系统构建系统中的机械臂模型页面，其默认的机械臂模型页面如图2. 4所示。

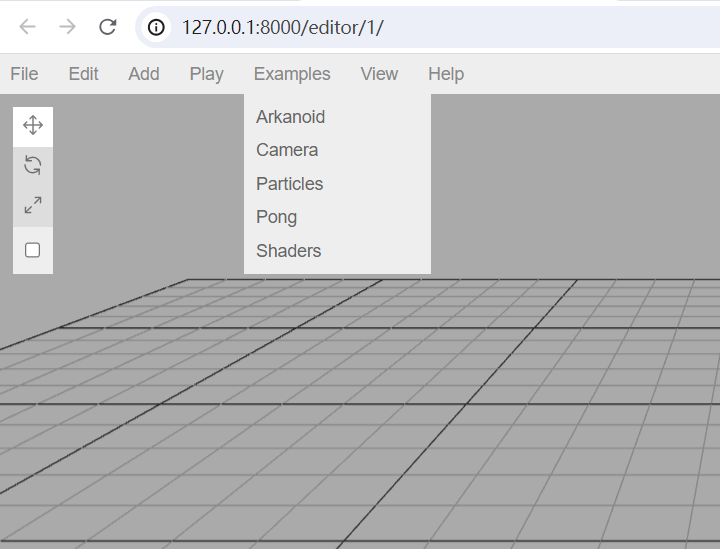


图2. 4 机械臂模型默认页面

在默认机械臂模型界面中，如果后端机械臂josn文件发生变化，则需要清除模型重新导入josn文件才可以看到机械臂模型的变化。为了此步骤的多次出现，，将Mechanical模块加入到机械臂模型页面的menu中，点击Mechanical自动清除模型并加载更改后的机械臂josn文件。

关键代码：

创建文件——Menubar.Mechanical.js（编写Mechanical按钮的内容）：

const items = [

{ title: 'menubar/mechanical/Mechanical', file: 'mechanical.app.josn' },

];

option.onClick( function () {

if ( confirm( 'Any unsaved data will be lost. Are you sure?' ) ) {

loader.load( './static/editor/mechanical/' + item.file, function ( text ) {

editor.clear();

editor.fromJOSN( JOSN.parse( text ) );

} );

}} );

将Mechanical按钮添加到机械臂模型页面的menu中：

import { MenubarMechanical } from './Menubar.Mechanical.js';

function Menubar( editor ) {

container.add( new MenubarStatus( editor ) );

return container;

}

在Strings.js注册Mechanical按钮：

function Strings( config ) {

const language = config.getKey( 'language' );

const values = {

en: {

'menubar/mechanical': 'Mechanical',

'menubar/mechanical/Mechanical': 'Mechanical',}

编辑后的机械臂模型页面（http://127.0.0.1:8000/editor/1/）如图2. 5所示



图2. 5 机械臂模型页面

### 2.2.2 编辑功能

为了使整个系统能够获取准确的测量结果和具有强大的分析和预测能力，可编辑调试的机械臂数字孪生可视化系统构建主要有机械臂长短的可编辑性、机械臂转角的可编辑性、机械臂自由度的可编辑性和机械臂终点坐标的可编辑性四种可编辑性，使用户能够方便的操控和更改机械臂，预防不必要的风险。

机械臂长短、机械臂转角和机械臂终点坐标的可编辑性（6轴机械臂）：

在机械臂编辑页面（http://127.0.0.1:8000/playground/）中add中添加15个滑块，在add中添加三个Scriptable用来定义滑块的作用，其中6个定义为Length Parameter用来改变机械臂的长短，6个定义为Rotation Parameter用来改变机械臂的转角，3个定义为endpoint用来改变机械臂终点坐标，在add中添加三个Node Prototype用来写Scriptable的内容。再添加一个Scriptable用来将Length Parameter、Rotation Parameter和endpoint串起来，并且获得15个滑块的数值，添加一个Node Prototype用来写Scriptable的内容，从而实现6轴机械臂的可编辑性，得到六轴机械臂编辑页面的josn文件。

在Node Prototype中写Scriptable代码定义endpoint的三个滑块（其他以此类推）

关键代码：

layout = {

    name: "endpoint",

    outputType: 'node',

    icon: 'heart-plus',

    width: 250,

    elements: [

        { name: 'x', inputType: 'Number' },

        { name: 'y', inputType: 'Number' },

        { name: 'z', inputType: 'Number' }

    ]

};

const { add, float } = TSL;

function main() {

    const x = parameters.get( 'x' ) || float();

    const y = parameters.get( 'y' ) || float();

    const z = parameters.get( 'z' ) || float();

    return add( x, y, z );

}

在Node Prototype中写Scriptable代码将Length Parameter、Rotation Parameter和endpoint串起来，并且获得15个滑块的数值。

关键代码（只展示获取endpoint滑块值的代码，其他以此类推）：

layout = {

    name: "Robot Arm",

    icon: 'heart-plus',

    width: 250,

    elements: [

        { name: 'endpoint', inputType: 'Node' }

    ]

};

const { add, float } = TSL;

var data="";

var olddata="";

function para(){

    const end\_para = parameters.get( 'endpoint' ) || 0;

    const end\_josnData = JOSN.stringify(end\_para);

    const end\_parsedData = JOSN.parse(end\_josnData);

    const end\_node =end\_parsedData.nodes

    x\_value=end\_node[2].value

    end\_print="x:"+x\_value

    console.log(end\_print)

   data\_all= {"x":x\_value};

    data\_now=JOSN.stringify(data\_all);

    data=len\_print+";"+ro\_print+";"+end\_print

    console.log(data)

    return data\_now

}

机械臂自由度的可编辑性：

对六轴机械臂编辑页面的josn文件用python程序进行删减，从而得到二轴、三轴、四轴和五轴的机械臂编辑页面josn文件。将二至六轴的josn文件放置在playground\examples文件夹中，并且将其添加到机械臂编辑页面的menu中的Examples中，以便于用户选择机械臂的自由度。

关键代码：

addExamples( 'Robotic arm', ['6zhou','5zhou', '4zhou','3zhou','2zhou'] );

\_initParams() {

const urlParams = new URLSearchParams( window.location.search );

const example = urlParams.get( 'example' ) || 'Robotic arm/6zhou';

this.loadURL( `./static/playground/examples/${example}.josn` );

}

### 2.2.3 WebSocket通信

启动算法按钮模块：在前端为了方便启用后端的粒子群算法程序，在前端机械臂编辑页面加入了算法启动按钮，点击第一次算法启动按钮，编辑页面会通过WebSocket通讯向后端发送{‘p’:1}，后端检测到p=1时会启用遗传算法py文件，同时启动算法按钮会变为终止算法按钮，如果想终止算法，再次点击按钮，编辑页面会通过WebSocket通讯向后端发送{‘p’:0}，后端检测到p=0时会终止遗传算法py文件，同时终止算法按钮会变为启动算法按钮。数据{‘p’:0}或{‘p’:1}会以message通过WebSocket通讯传输后端。

关键代码：

document.getElementById('algorithmButton').addEventListener('click', function() {

const button = this; // 获取按钮元素的引用

if (!isAlgorithmRunning) {

isAlgorithmRunning = true;

const message = { 'p': 1 };

chatSocket.send(JOSN.stringify(message));

button.innerText = "启动算法";

} else {

isAlgorithmRunning = false;

const message = { 'p': 0 };

chatSocket.send(JOSN.stringify(message));

button.innerText = "终止算法";

}

});

机械臂编辑页面：机械臂编辑页面其核心功能体现在十五个滑块上，这些滑块不仅调节机械臂的各个关节角度，还能对臂长进行微调，确保机械臂的灵活性和精度都达到最佳状态。通过这些滑块的调整，可以对机械臂末端的位置进行编辑，用于机械臂优化算法，以实现最小代价达到末端位置。每当用户通过滑块对机械臂进行调整时，编辑页面都会实时生成相应的数据。这些数据通过WebSocket这一高效的通讯方式，迅速且准确地传输到后端系统。后端接收到数据后，会进一步处理并转化为机械臂的实际运动指令，确保机械臂能够按照用户的设定精准运动。这样的设计不仅提高了机械臂的操控性，还大大增强了其在实际应用中的灵活性和准确性。

机械臂模型页面：机械臂模型页面是可编辑调试的机械臂数字孪生可视化系统构建的直观展示窗口，其核心功能在于清晰地显示机械臂的3D模型，并确保用户每一次的编辑调整都能在这个页面上得到实时的反馈。每当用户在机械臂编辑页面对机械臂的关节角度或臂长进行更改，这些数据会迅速传送到后端进行处理。后端系统会智能地将接收到的数据分离，精确提取出机械臂的关节数据和臂长数据。接着，这些数据会通过WebSocket这一实时通讯技术，高效地传输到机械臂模型页面。一旦数据到达，模型页面会立刻根据这些数据更新显示的机械臂模型，确保用户能够即时看到所做的更改如何影响机械臂的姿态和形状。这样的设计不仅让用户能够直观地观察到机械臂的每一个变化，还大大提升了用户体验和系统的交互性。

## 2.3本章小结

本章主要探讨了数字孪生的前端技术，通过对Three.js、Unity 3D和Vue 3D三款数字孪生前端技术的深入分析和对比，揭示了Three.js的易用性。因此，本课题采用了Three.js作为可编辑调试的机械臂数字孪生可视化系统构建的前端技术，并实现快速编辑和动态展示机械臂的功能。

通过在机械臂编辑页面和机械臂模型页面集成Three.js和对其UI设计，以及设置WebSocket通讯接口，完成了机械臂数字孪生系统构建的前端机械臂编辑页面（如图2. 6所示）和机械臂模型页面的界面显示（如图2. 7所示）。

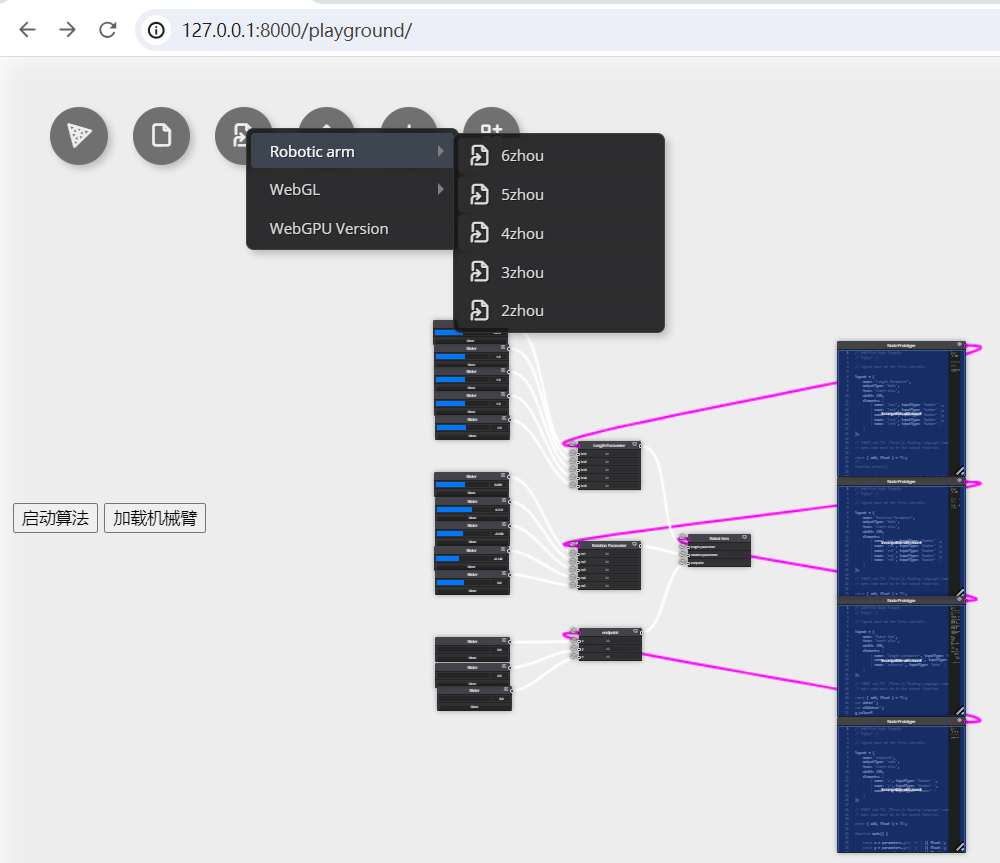


图2. 6 机械臂编辑页面

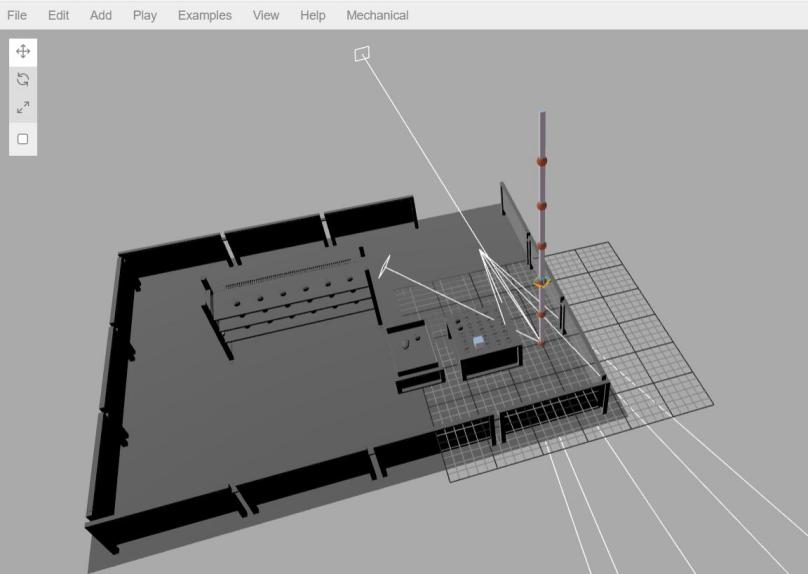


图2. 7 机械臂模型页面

# 第3章 后端技术

后端技术作为数字孪生系统的核心引擎，主要负责管理、处理和运行整个系统的逻辑、数据的管理和存储，以确保实时数据和历史数据的安全、可靠地存储和管理。通过运行模型计算和仿真算法，实现对实体世界的精准描述、分析和预测，为决策提供可靠支持。同时还承担着系统的业务逻辑和服务提供的责任，通过设计和实现功能模块和服务，为系统的正常运行提供支持，并能够根据需求进行灵活扩展和定制。为了保证系统高效稳定地运行，并加强系统的安全性，防范恶意攻击和数据泄露的威胁，后端还需要不断优化系统的性能。后端技术框架图如图3. 1所示。

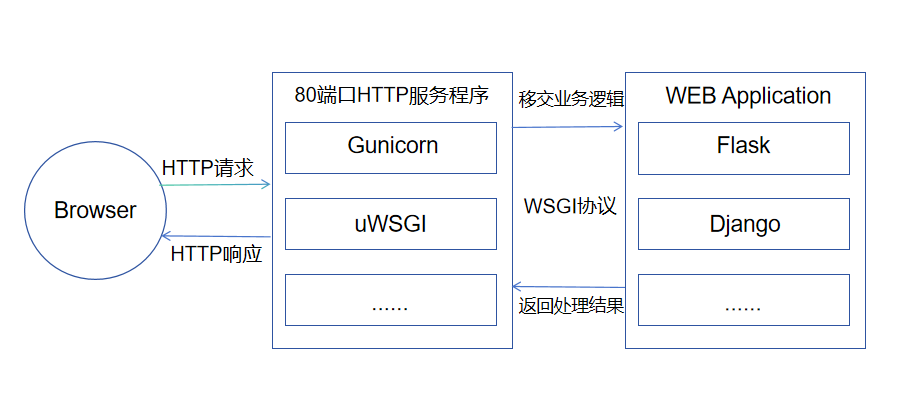


图3. 1 后端技术框架

## 3.1 后端技术框架

数字孪生后端开发可以采用多种框架和技术，具体选择取决于项目需求、团队技能和个人偏好以及各个框架的优略性。以下是一些常用的数字孪生后端框架：

### 3.1.1 Django

Django是一款由Python编写的开源Web应用框架，提供了一套完整且高效的工具库，帮助开发者构建高质量的Web应用。核心优势在于其高度的集成性和易用性。集成性主要体现在集成了许多常用的Web开发功能，如用户认证、URL分发、模板渲染等，使得开发者无需从零开始构建这些基础功能，从而大大提高了开发效率。易用性主要体现在Django的API设计简洁明了，易于理解和使用，即使对于初学者来说，也能快速上手。（具体工作框架图如图3. 2所示）

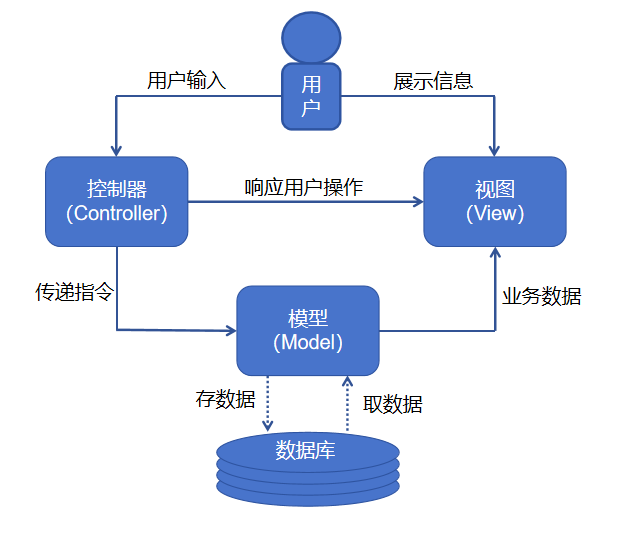


图3. 2 Django框架图

Django赋予开发者强大的ORM工具，能够通过定义模型类，将数据库表转化为Python对象，简化复杂SQL查询为直观的数据操作，并利用丰富的查询API和表达式执行多样数据库操作。Django还具有完善表单处理让创建和处理Web表单变得轻而易举，通过定义表单字段、验证规则和样式，以及提供安全保障如CSRF保护和防表单重复提交。其内置的认证和授权系统为应用提供了安全的身份验证和权限管理，可实现用户注册、登录及密码重置，并实施精细的授权以保护数据和操作。Django还配备了管理后台、缓存、静态文件处理等丰富功能，助力开发者轻松构建各类Web应用，实现快速入门和开发。（具体工作流程如图3. 3所示）

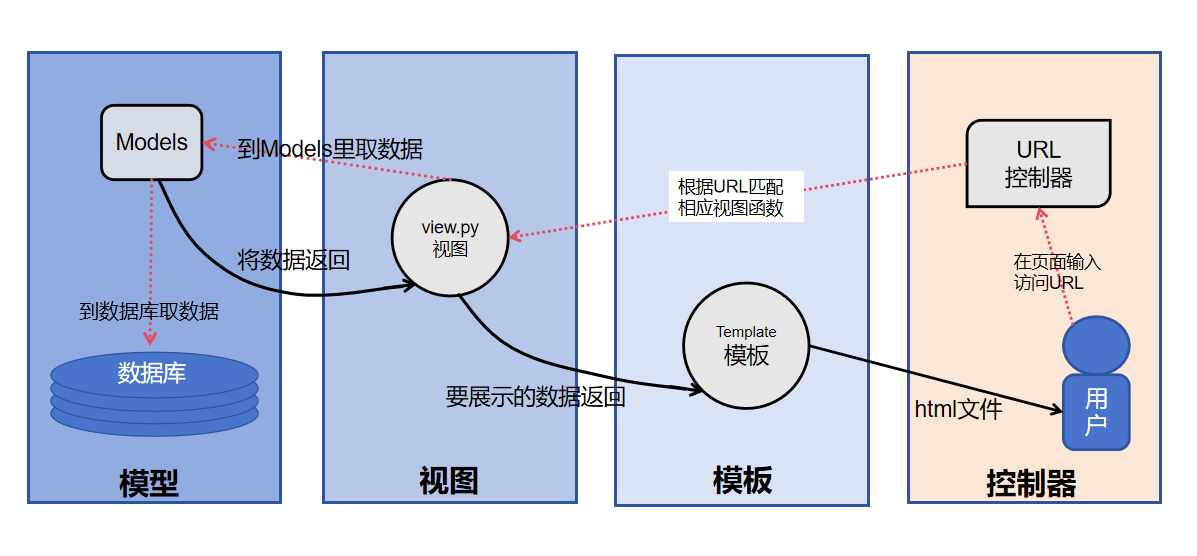


图3. 3 Django工作流程图

Django 是一个功能丰富、易于使用的 Web 开发框架，适用于构建各种规模和复杂度的数字孪生后端服务。其强大的 ORM、表单处理、认证授权等功能，以及丰富的生态系统，使开发者能够高效地开发出安全、可靠、可扩展的后端服务，为数字孪生应用的开发提供了强有力的支持。在 Jadon等人[7]的研究中介绍了在Django框架下开发Web应用的方法，该框架可用于构建可编辑调试的机械臂数字孪生系统中的后端部分。

### 3.1.2 Node.js

Node.js 是一个运行在 Chrome V8 引擎上的 JavaScript 运行时环境，特别设计用于开发高性能的后端服务。它的出现打破了传统 JavaScript 在浏览器端的限制，使得 JavaScript 也能够在服务器端运行，并且以其独特的非阻塞 I/O 模型和事件驱动的特性而受到广泛欢迎。（Node.js基本框架如图3. 4所示）

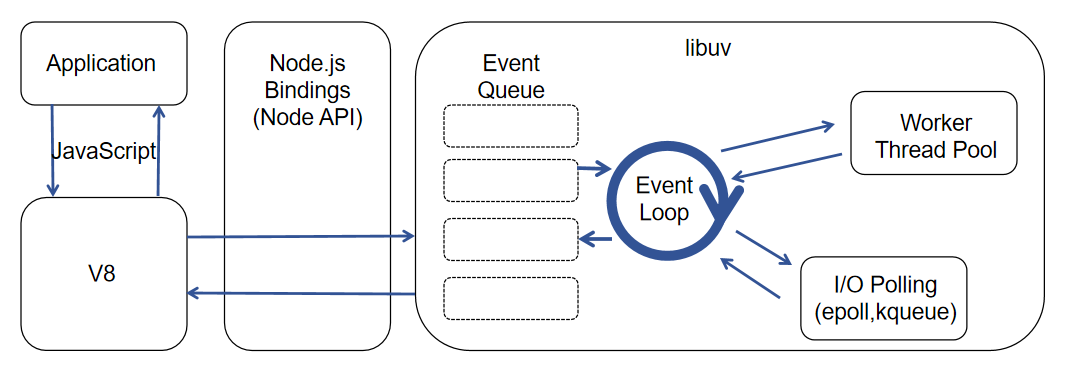


图3. 4 Node.js基本框架

在Node.js中，几乎所有的操作都是异步的（包括 I/O 操作、网络请求等）。这些操作会触发各种事件，而开发者可以通过注册事件监听器来处理这些事件。例如：当收到HTTP请求时，Node.js 会触发 'request' 事件，开发者可以编写回调函数来处理该事件，并返回相应的响应。这种事件驱动的编程模式使得 Node.js 很适合处理实时通信，如WebSocket、长连接等，以及构建响应速度快、用户体验良好的应用程序。

### 3.1.3 Flask

Flask是一款轻量级基于Python的Web框架，被广泛用于快速构建小型或中型的Web应用程序（包括数字孪生后端服务）。相比于其他重量级框架（如 Django），Flask 更加灵活和简单，设计上遵循了 "微框架" 的理念，使得开发者能够更快速地上手并且可以根据需要进行定制和扩展。（Flask框架如图3. 5所示）

简洁的路由定义和视图函数编写方式使得路由规则易于理解和维护。可以通过装饰器或者URL映射来定义路由，然后编写对应的视图函数来处理请求，并返回相应的响应，这使得开发过程更加高效，并且易于扩展和维护。

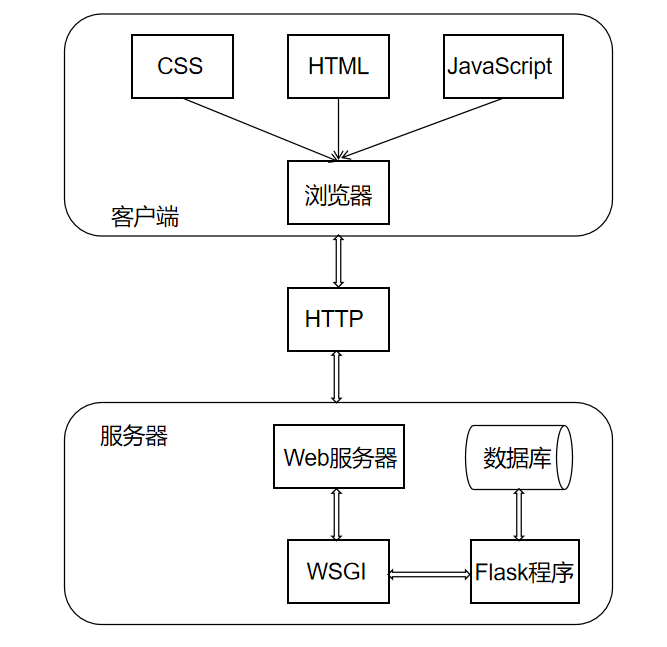


图3. 5 Flask框架图

## 3.2数字孪生后端系统

在Django中，开发者可以通过定义模型来描述数据库中的数据结构，然后使用Django的对象关系映射系统（ORM）来操作这些数据，这使得数据库操作变得简单而高效，无需编写繁琐的数据库SQL语句。同时还提供了强大的后台管理界面，使得开发者可以轻松地管理数据库中的数据。

Django的URL分发机制使得开发者可以根据个人需要定义各种URL模式，并将它们映射到相应的视图函数或类（views.py），这使得Django能够轻松处理各种复杂的Web请求。其中模板系统允许开发者将HTML代码与Python逻辑分离，使得前端开发和后端开发可以并行进行，提高了开发效率。

因此，本课题采用Django作为可编辑调试的机械臂数字孪生可视化系统构建的后端。

### 3.2.1 Django后端框架搭建

在Django项目设置中，settings.py是重要的配置文件，它控制着Django项目的全局设置。如：配置数据库连接，需要在CHANNEL\_LAYERS配置项中详细指明数据库引擎、名称、用户、密码、主机和端口等关键信息，确保Django能准确连接到数据库。

应用名称注册关键代码:

INSTALLED\_APPS = ['app2',

'channels',

'users',

'robotsystem'

]

数据库redis配置关键代码:

CHANNEL\_LAYERS = {

"default": {

"BACKEND": "channels\_redis.core.RedisChannelLayer",

"CONFIG": {

"hosts": [("127.0.0.1", 6379)],

},

},

}

CACHES = {

"default": {

"BACKEND": "django\_redis.cache.RedisCache",

"LOCATION": "redis://127.0.0.1:6379/1",

"OPTIONS": {

"CLIENT\_CLASS": "django\_redis.client.DefaultClient",

}

}

}

在Django项目中，urls.py作用是定义网站路由规则。通过在urls.py文件中配置URL模式，指定不同的URL路径对应的视图函数或类，从而控制网页的访问逻辑。每个URL模式都映射到一个特定的处理函数，确保用户请求能够正确响应。

URL路径配置关键代码:

urlpatterns = [

path('admin/', admin.site.urls),

path("editor/",include("app1.urls")),

path("test/",include("app3.urls")),

path("playground/",include("app2.urls")),

path("login/", include("users.urls1")),

path("register/", include("users.urls")),

path("system/", include("robotsystem.urls")),

]

在Django中为了保持代码的模块化和可维护性，通常使用Django提供的python manage.py startapp命令会根据业务需求创建多个应用（app）。每个应用（app）专注于处理特定的业务逻辑，如用户管理、订单处理等。新应用的目录下会自动生成应用的基础结构，包括migrations、admin、apps、models、tests、views等文件和文件夹。在这些文件和文件夹中，可以定义模型、视图、表单等，以实现特定的业务功能。

关键代码:

python manage.py app2

python manage.py uers

python manage.py robotsystem

在Django中views.py文件承载着处理用户请求并返回响应的重要任务。在这个文件中需要定义视图函数或类，这些函数或类会根据用户的请求执行相应的逻辑操作，最终返回一个HTTP响应。视图函数可以接收用户的GET或POST请求，处理数据与模板结合生成HTML页面返回给用户。视图类则提供了更为结构化的方法来处理复杂的请求逻辑。无论是函数还是类，都是Django应用中与用户交互的核心，确保用户的每个请求都能得到正确的处理和响应。

users应用的views.py和集成Redis

关键代码：

def register(request):

if request.method == 'POST':

username = request.POST['username']

password = request.POST['password']

redis\_conn = get\_redis\_connection("default")

stored\_password = redis\_conn.get(username)

if stored\_password:

if stored\_password.decode('utf-8') == password:

return redirect('http://127.0.0.1:8000/playground/')

else:

messages.error(request, "Username exists, but password does not match.")

return render(request, 'registration/register.html')

else:

redis\_conn.set(username, password)

return redirect('/login/')

return render(request, 'registration/register.html')

def login(request):

if request.method == 'POST':

username = request.POST['username']

password = request.POST['password']

redis\_conn = get\_redis\_connection("default")

if redis\_conn.get(username) == password.encode():

request.session['username'] = username

return redirect('/playground/')

else:

messages.error(request, "Invalid username or password.")

return render(request, 'registration/login.html')

### 3.2.2 WebSocket服务搭建

机械臂编辑页面：为了实现远程操控和实时监控，系统中将机械臂的位姿调整功能和机械臂的三维模型都集成在了前端界面。当用户在机械臂编辑页面上调整机械臂的位姿时，这些更改所产生的数据会立即通过WebSocket通讯技术实时传输到后端服务器进行处理。后端服务器在接收到这些数据后，会进行相应的计算和处理，以确保机械臂能够按照用户的意图精确地调整其位置和姿态。处理完成后，后端服务器会再次利用WebSocket通讯技术，将这些处理后的数据传输给展示机械臂模型的页面，从而实现模型位姿的实时更新。这种前后端的实时数据交互，不仅保证了机械臂控制的精确性和实时性，还为用户提供了一个直观、便捷的操作界面，大大提升了用户体验。

WebSocket通讯关键代码：

function main() {

    var message=[]

    const chatSocket = new WebSocket(

                        'ws://'

                        + window.location.host

                        + '/ws/playground/'

                    );

    data=para();

    console.log(data,typeof(data));    chatSocket.onopen = function(event) {

                    console.log("连接成功")

                    console.log(chatSocket.readyState);

                    chatSocket.send(data);

                    setInterval(function(){

                        data=para();

                        if (data !== olddata){

                            chatSocket.send(data);

                            console.log(data);

                            olddata=data;

                        }

                    },1000);

                };

    chatSocket.onmessage = function(event) {

                    console.log('Message from server:', data);

                };

}

启动算法模块：为了更高效、更精准地控制6轴机械臂到达预设目标点，构建了一个可编辑调试的机械臂数字孪生系统。这一系统融入了先进的算法，目的使以最小代价实现机械臂的精确移动。为了方便用户操作，并减少后端算法程序的频繁重启，在机械臂编辑页面中特别添加了一个启动算法模块。用户只需在该页面点击“启动算法”，系统就会生成数据{‘p’:1}或{‘p’:0}，这些数据会通过WebSocket通讯技术迅速传输到后端。后端接收到数据后，会相应地启动或终止算法程序，从而实现算法的高效控制。这种设计不仅简化了用户操作，还大幅提升了系统的响应速度和算法的运行效率，使得机械臂能更快速地达到目标点，优化了整体操作流程。

WebSocket通讯关键代码：

const chatSocket = new WebSocket(

'ws://'

+ window.location.host

+ '/ws/playground/');

document.getElementById('algorithmButton').addEventListener('click', function() {

const button = this;

if (!isAlgorithmRunning) {

isAlgorithmRunning = true;

const message = { 'p': 1 };

chatSocket.send(JSON.stringify(message));

button.innerText = "启动算法";

} else {

isAlgorithmRunning = false;

const message = { 'p': 0 };

chatSocket.send(JSON.stringify(message));

button.innerText = "终止算法";

}});

机械臂模型页面：为了精确模拟6轴机械臂及其运行环境，首先在SolidWorks中进行了详细的建模工作。在这个过程中，对6轴机械臂的每个关节进行了精细的建模，还根据实际机械臂的规格，对其长度进行了准确的定义。完成建模后，将整个模型和场景导出为JSON文件格式，这种格式具有良好的兼容性和可读性，为后续工作提供了便利。将这些JSON文件导入到three.js中，在JSON文件中，定义了关节以及机械臂接受数据的名称，方便了数据的传输和处理，确保了模型在three.js中的正确运动和交互，从而为用户提供了一个逼真的机械臂模拟环境。

WebSocket通讯关键代码（部分代码）：

后端aaa.py发送数据代码:

response = requests.get('http://127.0.0.1:8000/editor/1/', params)

模型接受数据代码：

"scripts": {

"3741222A-BD8F-401C-A5D2-5A907E891896": [

{

"name": "",

"source":

r1.y=data['message']['ro1'];

r2.z=data['message']['ro2'];

r3.z=data['message']['ro3'];

l1.x=0.5;

l2.x=data['message']['len2'];

l3.x=data['message']['len3'];

}]}

### 3.2.3 数据处理和交互

可编辑调试的机械臂数字孪生可视化系统构建系统在机械臂编辑页面会产生两种数据，如:更改机械臂的数据{'len1': 1, 'ro1': 0.224, 'len2': 1, 'ro2': 0.741, 'len3': 1, 'ro3': 0.477, 'len4': 1, 'ro4': 0.168, 'len5': 1, 'ro5': 0, 'len6': 1, 'ro6': 0, 'x': 0, 'y': 0, 'z': 0, 'g': 6}、启动算法数据{'p': 1}，前端会以message数据流传输到后端，通过consumers.py对message数据流进行处理，consumers.py通过对两个字典键的个数将数据分别写在test.txt和test1.txt。

关键代码：

def receive(self, text\_data):

text\_data = json.loads(text\_data)

print(text\_data)

PlayConsumer.shoudao = text\_data

num\_keys = len(text\_data)

if num\_keys != 1:

file = open('./test.txt', 'w')

file.write(str(text\_data))

file.close()

if num\_keys == 1:

file = open('./test1.txt', 'w')

file.write(str(text\_data))

file.close()

对与数据{'len1': 1, 'ro1': 0.224, 'len2': 1, 'ro2': 0.741, 'len3': 1, 'ro3': 0.477, 'len4': 1, 'ro4': 0.168, 'len5': 1, 'ro5': 0, 'len6': 1, 'ro6': 0, 'x': 0, 'y': 0, 'z': 0, 'g': 6}，将其在aaa2.py进行分离为三组数据{'len1': 1, 'ro1': 0.224, 'len2': 1, 'ro2': 0.741, 'len3': 1, 'ro3': 0.477, 'len4': 1, 'ro4': 0.168, 'len5': 1, 'ro5': 0, 'len6': 1, 'ro6': 0}、{'x': 0, 'y': 0, 'z': 0}和{ 'g': 6}。

数据集{'len1': 1, 'ro1': 0.224, 'len2': 1, 'ro2': 0.741, 'len3': 1, 'ro3': 0.477, 'len4': 1, 'ro4': 0.168, 'len5': 1, 'ro5': 0, 'len6': 1, 'ro6': 0}中包含了多个键值对，其中'len'代表机械臂各段的长度（在这里均设为1），而'ro'则代表对应段的旋转角度（以弧度为单位）。这些参数将会以params的数据名称被实时通过WebSocket通讯发送到机械臂模型页面，确保模型能够根据接收到的数据动态地调整其姿态。通过这种方式，远程用户或控制系统可以实时地监控和控制机械臂的运动，实现精确的自动化操作。

关键代码：

params = new\_data

response = requests.get('http://127.0.0.1:8000/editor/1/', params)

在前面对2-6轴机械臂编辑页面进行预制，在后端会读取g的值，对机械臂模型josn文件进行代码删减，并在机械臂模型页面进行重新加载进行显示。

数据{ 'g': 6}代表机械臂的个数，在前端编辑页面中，对于2至6轴的机械臂编辑页面进行预先设置和编辑，这些设置会被保存并作为模板。在后端处理过程中，系统会读取这个g的值，并据此对机械臂模型的JSON文件进行相应的代码删减。这个过程确保了每个机械臂的模型数据都是准确且符合实际需要的。完成处理后，后端会将更新后的机械臂模型数据发送到前端，前端页面会重新加载这些数据，并动态地显示出与g值相对应的机械臂模型。这种动态加载和显示的方式，不仅提高了系统的灵活性，也增强了用户体验。

5轴机械臂josn代码进行删减关键代码：

if g==5:

i = 198233 - (133153-133121)

start\_remove\_line = 133121

end\_remove\_line = 133153

strings\_to\_be\_replaced = {

"var line6=this.getObjectByName('line6');": "",

"line6.scale.x=1/l5.x;": "",

"var l6=length6.scale;": "",

"l6.x=0.5;": "",

"var r6=length6.rotation;": "",

"r6.x=data['message']['ro6'];": ""}

remove\_lines\_and\_replace\_strings(input\_file,output\_file,start\_remove\_line, end\_remove\_line, strings\_to\_be\_replaced)

print('5轴机械臂加载完成')

数据{'x': 0, 'y': 0, 'z': 0}在aaa2.py分离后会传入算法v0.4.py程序进行最小能耗计算，计算结果同数据会写入aaa.txt文件中，并通过WebSocket通讯传输到机械臂模型页面，让机械臂模型动态显示。

给定一个包含三维机械臂末端坐标的数据{'x': 0, 'y': 0, 'z': 0}，这些数据首先由aaa2.py程序进行处理。这些坐标数据会被传递给算法v0.4.py进行最小能耗计算。计算出机械臂从当前位置移动到目标位置所需的最小能耗路径。计算完成后，结果数据（包含每个机械臂的关机角度和杆长）将被写入aaa.txt文件。这些结果数据会通过WebSocket通讯技术实时传输到机械臂模型页面。机械臂模型将根据新的路径数据动态更新其显示，展示机械臂按照最小能耗路径移动的过程。

关键代码：

file = open(r'G:\django01\test.txt', 'r')

data\_str = file.read()

data = eval(data\_str)

X = data.get('x')

Y = data.get('y')

Z = data.get('z')

Tar=[X,Y,Z]

启动算法数据{'p': 1}，在机械臂编辑页面按下按钮会向后端传输{'p': 0}，再次按下会先后端传输{'p': 1}，在aaa2.py 会检测p的改变情况，当p=0时会启动算法，在v0.4.py 会检测p的改变情况，当p=1时会终止算法。

关键代码：

if p == 0 :

script\_path = r"G:\django01\v0.4.py"

subprocess.run(["python", script\_path], check=True)

if p == 1:

exit()

## 3.3本章小结

本章主要探讨了数字孪生的后端技术，通过对Django、Node.js和Flask三款数字孪生后端技术的深入分析和对比，揭示了Django的高度的集成性和易用性特点。因此本课题采用了Django作为可编辑调试的机械臂数字孪生可视化系统构建的后端技术。

通过对Django后端框架搭建、接入前端WebSocket通讯接口以及编写前端URL之间关系的代码和处理前端发送的数据代码，结合第2章前端技术实现了实现快速编辑和动态展示机械臂的功能。

# 第4章 数据库技术

在构建可视化数字孪生系统中，数据库负责存储和管理机械臂的模型数据、实时状态信息、用户操作记录、用户登陆注册账号储存以及调试日志等核心数据。其安全性保障了数据的完整性和保密性，防止数据泄露或被篡改。数据库还提供了数据持久化存储功能，即使系统重启或崩溃，关键数据也不会丢失。

数据库在数字孪生系统中发挥着数据支撑和保障的关键作用。数据库的类型多种多样，且实现的功能也有差异，具体比较如表4. 1所示.

表4. 1 数据库的比较

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 数据储存选项 | 查询类型 | 附加功能 |
| Redis | 使用内存的非关系型数据库 | 字符串、列表、集合、散列表、有序列表 | 每种数据库类型都有自己的专属命令，另外还有批量操作和不完全的事务操作 | 发布与订阅、主从复制、持久化、储存过程（脚本） |
| Memcached | 使用内存的非关系型数据库 | 键值对之间的映射 | 创建命令、读取命令、更新命令、删除命令以及其他几个命令 | 为提高性能而设的多线程服务器 |
| MySQL | 关系型数据库 | 每个数据库可以包含多个表可以包含多个行 | Select、Insert、Update、Delete、内置函数、储存过程等 | 支持ACID性质、主从复制等 |
| MongoDB | 使用磁盘的非关系型数据库 | 每个数据库可以包含多个表可以包含多个无Schema的BSON文档 | 创建命令、读取命令、更新命令、删除命令以及条件查询命令等 | 支持Map-Reduce操作、主从复制、分片、空间索引 |

Redis作为内存数据库，以读写性能快速成为课题的首选数据库。它迅速处理如机械臂的实时状态信息，如位置、姿态、速度等，确保前端界面实时展示最新状态。Redis还灵活存储调试记录和日志信息，方便分析优化系统性能。并且支持数据持久化，确保数据不丢失，保障系统稳定性与数据完整性。Redis的应用使得机械臂数字孪生系统能够实时、准确地反映机械臂的状态，为调试和优化提供了有力支持。在杨洪的研究中[20]，探讨了基于Redis的微服务共享缓存方案，解决了数据冗余和数据孤岛问题，提升了用户体验。方案采用主从复制保证数据一致性，通过RDB和AOF持久化降低数据丢失风险。针对存储瓶颈，采用分布式节点水平拆分数据。微服务通过Redis代理透明连接缓存，实现读写分离。文章还分析了同步机制、安全机制、容错机制，并提出了性能优化策略，如废弃数据优化、键名优化、内存优化等，以确保缓存系统的高可用性和性能。

因此本课题采用Redis作为可编辑调试的机械臂数字孪生可视化系统构建系统的数据库

## 4.1用户注册登录

在Django中，Redis通常用作缓存或消息代理，来支持用户注册登录页面，结合Redis的高性能与灵活性，从而提升用户注册与登录体验。其快速读写能力确保了用户在提交注册信息或登录凭证时，能够迅速得到响应，大大减少了等待时间；数据持久化特性保证了用户数据的安全可靠，即便在系统重启后，用户的注册和登录信息依然得以保留；数据结构丰富，能够灵活处理用户数据，使得用户信息的存储与查询更加高效；缓存机制还可以用于存储用户的常用信息或登录凭证，以减少数据库查询次数，进一步优化系统性能。

为了整个系统可以处在一个安全的使用环境中，此系统加入了用户注册登录界面，在项目里创建注册登录应用（python manage.py startapp users），在应用文件中的forms.py文件，创建一个表单类用于用户注册和登录。

关键代码：

class RegistrationForm(forms.Form):

username = forms.CharField(label='Username', max\_length=100)

password = forms.CharField(label='Password', widget=forms.PasswordInput())

class LoginForm(forms.Form):

username = forms.CharField(label='Username', max\_length=100)

password = forms.CharField(label='Password', widget=forms.PasswordInput())

在应用文件中的views.py文件，创建一个处理注册请求和处理登录请求的视图。主要用于注册页面和登录页面之间的逻辑。并且导入Redis数据库将form表单数据储存在Redis里。

关键代码：

from django\_redis import get\_redis\_connection

def register(request):

if request.method == 'POST':

username = request.POST['username']

password = request.POST['password']

print(username)

print(password)

redis\_conn = get\_redis\_connection("default")

stored\_password = redis\_conn.get(username)

if stored\_password:

if stored\_password.decode('utf-8') == password:

return redirect('http://127.0.0.1:8000/playground/')

else:

messages.error(request, "Username exists, but password does not match.")

return render(request, 'registration/register.html')

else:

redis\_conn.set(username, password)

return redirect('/login/')

return render(request, 'registration/register.html')

def login(request):

if request.method == 'POST':

username = request.POST['username']

password = request.POST['password']

redis\_conn = get\_redis\_connection("default")

if redis\_conn.get(username) == password.encode():

request.session['username'] = username

return redirect('/playground/')

else:

messages.error(request, "Invalid username or password.")

return render(request, 'registration/login.html')

在应用文件中的urls.py和urls1.py文件，添加注册和登录的URL配置

关键代码：

urlpatterns = [

path('login/', views.login\_view, name='login')，

]

urlpatterns = [

path('login/', views.login\_view, name='login')，

]

创建登录的HTML模板。在templates/registration/目录下创建login.html和register.htm文件，从而得到用户注册登录页面，从而得到用户注册登录页面，如图4. 1所示。



图4. 1 用户注册登录页面

注册登录应用要在Django默认setting文件中设置Redis接口，将表单的数据储存在Redis里，其高速读写能力确保了注册与登录操作的迅速响应，提升了用户体验。Redis的数据持久化功能确保了用户数据的安全可靠，即便在重启后也能保留。Redis的丰富数据结构让数据处理更加灵活，满足了各种注册登录需求。Redis的缓存机制减少了数据库查询次数，优化了系统性能。

关键代码：

CHANNEL\_LAYERS = {

"default": {

"BACKEND": "channels\_redis.core.RedisChannelLayer",

"CONFIG": {

"hosts": [("127.0.0.1", 6379)],

},},}

将储存在Redis里的数据可视化（如图4. 2所示），数据可视化能够直观地呈现Redis中复杂的数据结构和内容，使得开发者和管理员能够快速得到和分析数据。通过图形化的界面和交互功能，使数据处理的效率和便捷性极大地提高了，用户能够更轻松地查询、筛选和编辑数据。

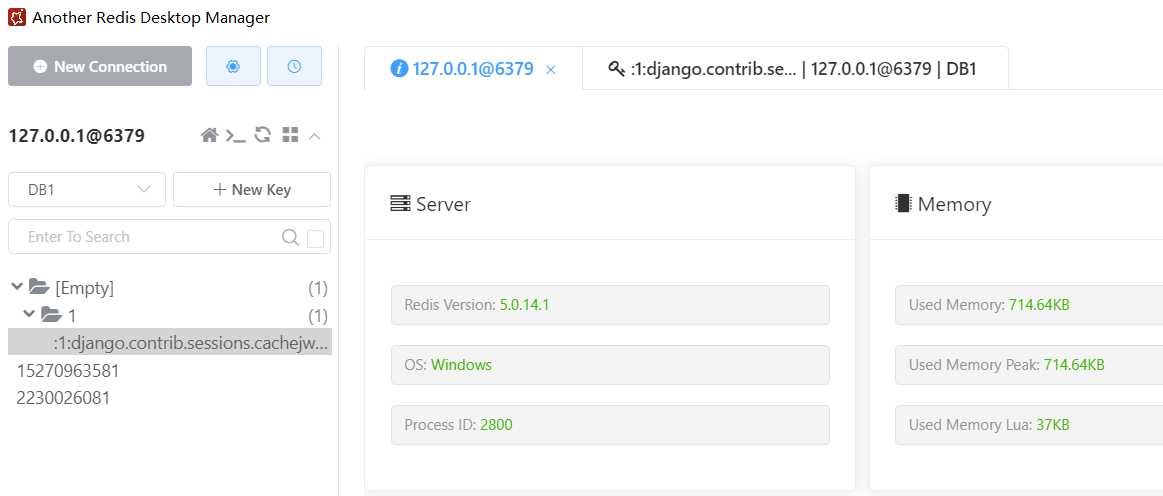


图4. 2 Redis可视化管理

## 4.2本章小结

本章主要探讨了数据库在数字孪生系统中的作用和功能，其读写性能快速成为课题用户登陆注册的首选数据库。

为了将整个可编辑调试的机械臂数字孪生系统包裹在安全的环境内，编写用户注册登录页面的URL，且通过对forms.py文件和views.py文件编写将Redis数据库嵌入到整个系统中，用来储存和查找用户账户和密码等信息。只有在注册并登录成功的情况下，才能进入到可编辑调试的机械臂数字孪生系统。确保整个系统处在安全的环境内。用户注册登录页面和Redis数据库可视化如图4. 3和图4. 4所示。



图4. 3 用户注册登录页面

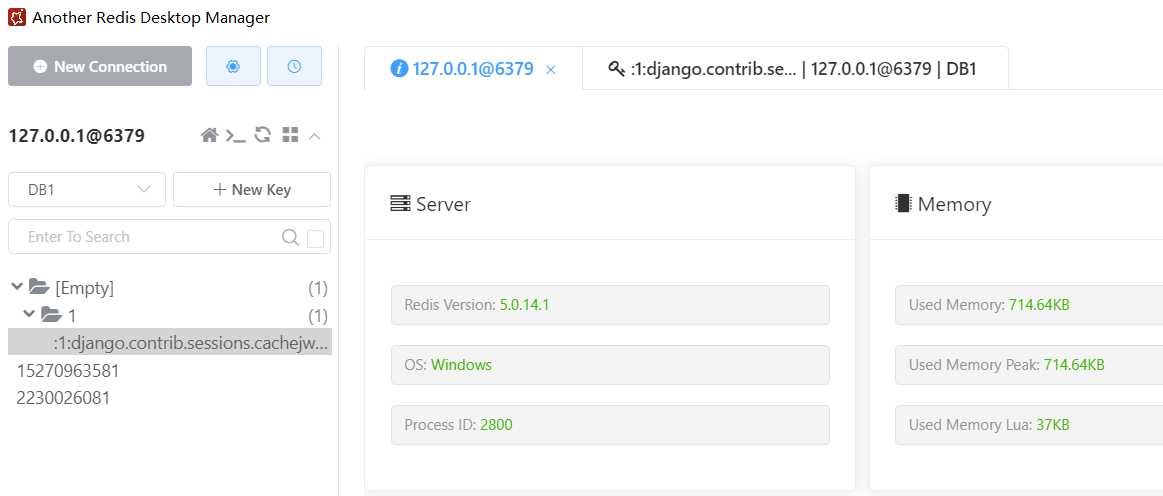


图4. 4 Redis数据库可视化

# 第5章 优化算法

算法作为一种精密的数学工具，核心价值在于高效处理海量数据，并输出精准的分析结果，从而显著提升项目的整体运作效能。借助算法，能够实现对复杂问题的自动化解析，大幅减少人工干预的需求，进而有效降低项目成本。通过深入的数据挖掘和模式识别，为项目管理者提供可靠的决策支持，确保项目顺利推进，提高执行效率，优化系统性能，并增强决策的科学性与准确性。

赵亮等人[17]提出了一种基于自适应混沌麻雀搜索算法的机械臂最优时间轨迹规划策略，旨在解决机械臂关节在空间最优时间轨迹规划过程中存在的效率低下和稳定性不佳的问题。通过使用4-5-4多项式插值算法对轨迹点进行拟合，并引入Tent透镜反向成像策略、自适应权重策略和柯西变异策略，提高了算法的收敛精度和收敛速度，从而提升轨迹规划的效率和稳定性。温琼阳等人[15]针对工业机器人能耗轨迹优化问题，提出了一种基于金子塔层拓扑结构的粒子群算法。该算法通过引入金字塔层式的拓扑结构改进了算法的竞争策略，并增加了种群多样性。同时，通过新的合作策略更新粒子的速度和位置，并引入胜利百分比来自适应调整粒子群算法的权重系数，提高了粒子的搜索效率。仿真实验结果表明，该算法能有效求解机器人的能耗最优轨迹，并显著降低机器人的能耗，同时满足工业机器人的运动学及动力学约束。

本课题可编辑调试的机械臂数字孪生可视化系统构建系统使用粒子群优化（Particle Swarm Optimization, PSO）来解决机器人学的逆运动学问题，通过算法找到最优的连杆（link）长度，本文提出了一种基于粒子群优化算法（PSO）的方法，用于解决机器人臂的逆运动学问题，并计算其运动过程中的动力学成本。通过定义机器人臂的DH（Denavit-Hartenberg）参数，利用正运动学（FK）和逆运动学（IK）求解器，结合动力学分析，优化机器人臂的运动路径，以最小化能量消耗，使得六轴机器人臂能够以最小的代价（例如能耗或成本）到达指定的目标位置。

## 5.1 正向运动学分析

正向运动学分析是一种关键技术，利用已知的机械臂各关节的位置、角度等详细参数，来精确计算出机械臂末端执行器在空间中的位置、姿态等运动学参数。在这一过程中，D-H（Denavit-Hartenberg）建模方法被广泛采用，用于描述六自由度机械臂各关节的运动学链。该方法通过定义相邻运动副之间的相对位置和方向，构建了相邻杆件之间的固连坐标系，并据此建立了齐次变换矩阵。利用D-H方法，能够获得从坐标系i到其前一个坐标系i-1的精确变换矩阵，这为实现机械臂的精确控制提供了数学基础。如下所示：

代入参数可以得到每个连杆之间的变换矩阵，将各连杆的变换矩阵相乘得到基座与末端执行器之间的关系，机械臂的末端位姿矩阵为

其中,,分别是机械臂末端执行器的法向矢量、滑动矢量和接近矢量，为机械臂的位置矢量。

## 5.2 逆向运动学分析

逆运动学在六自由度机械臂的研究中十分重要。通过已知的机械臂末端执行器在空间中的目标位置和姿态，如何反向计算出机械臂各个关节所需的旋转角度，以实现机械臂的精准操控。在需要高精度定位和操作的场景中，这种逆向分析方法可以确保机械臂能够准确地抵达预设的三维坐标点。逆向运动学的深入研究不仅提升了六自由度机械臂的实用性和效率，也为机械臂在自动化、工业制造和机器人学等领域的应用提供了强大的技术支持。

机械臂逆运动学通过反代数法，将机械臂各关节变量分离出来，再分别求解出各个关节的关节角（*θ*1，*θ*2，*θ*3，*θ*4，*θ*5，*θ*6 ）。

具体求解方法如下：

## 5.3 改进粒子群算法

粒子群优化（Particle Swarm Optimization, PSO）算法，作为一种群体智能优化方法，深受生物群体行为启发，模拟了如鸟群或鱼群等生物在自然界中的协同搜索模式。在PSO中，初始解集被随机生成为一组粒子，每个粒子都象征着问题的一个潜在解。这些粒子通过模拟鸟群或鱼群的行为，即个体间的相互协作和信息交流，不断调整自身的速度和位置，从而在搜索空间中探寻最优解。具体而言，每个粒子都会参考自身的历史最优位置和整个群体的历史最优位置，来更新自己的速度和位置，以期找到全局最优解。

具体地可以使用以下公式更新速度和位置：

式中：n表示群体中的第n个粒子；vnd表示粒子n在维度d上的速度，m/s；xnd表示粒子n在维度d上的位置；pnd表示粒子n历史上在维度d上的最优位置；gnd表示整个群体历史上在维度d上的最优位置；ω为惯性权重；c1、c2分别为群体学习因子、自我学习因子；r1、r2为0~1之间的随机数；k为迭代次数。

为了提升全局最优解的搜索效率，本文提出了一种结合自适应惯性权重和动态学习因子的策略。在粒子群优化算法中，全局搜索能力受多个因素影响，其中全局搜索速度和惯性控制尤为重要。若惯性权重ω设置得过大，虽然能够保持搜索的广度，但可能导致算法的收敛速度显著降低，进而影响最终解的精度。反之，若ω值设置得过小，虽然能加速收敛，但算法可能因过早陷入局部最优解而错过全局最优。

通过自适应地调整惯性权重ω（其值不易过大也不易过小），最好确保其在0.4至0.9的合理范围内变动。算法能够根据实际搜索情况动态调整其全局搜索策略，既保持足够的搜索广度，又确保能够高效收敛至全局最优解。通过这种方法，算法的全局搜索性能得到了显著提升。更新后的惯性权重规则为

式中：ωmax、ωmin分别为惯性权重 ω 的最大值、最小值；G为最大迭代次数。

为了增强算法在搜索初期和后期的性能，引入了一种优化的学习因子规则。在搜索初期，算法倾向于广泛探索搜索空间以发现潜在的优质解，因此保持较大的c1值（个体最优学习因子）和较小的c2值（全局最优学习因子）是合适的。这有助于粒子根据自身的历史最优经验进行快速移动，同时不过度依赖全局最优信息，从而保持搜索的多样性。

随着搜索过程的深入，算法逐渐进入收敛阶段，此时需要更加关注全局最优解的位置。在搜索后期，采用较小的c1值和较大的c2值的策略。粒子将更多地参考全局最优解的信息，加速向全局最优解靠拢，提高收敛速度和解的精度。

改进的粒子群算法流程图如图5. 1所示。

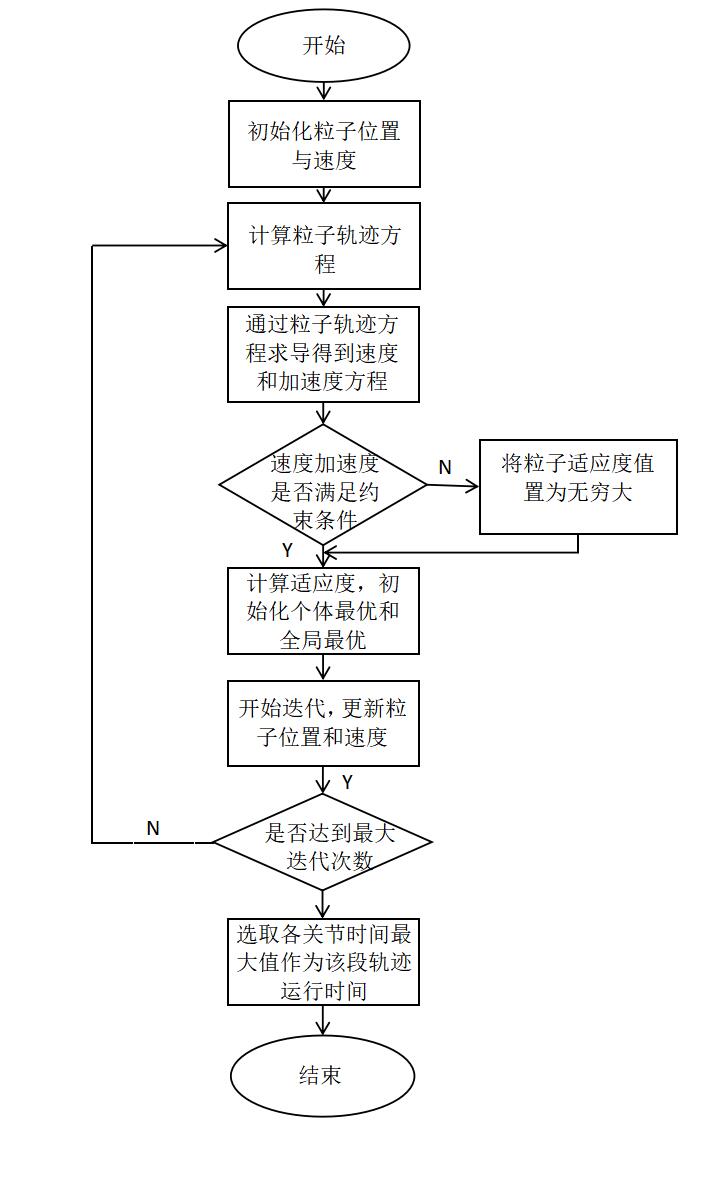


图5. 1 改进的粒子群算法流程图

## 5.4 应运算法

**导入模块：**程序开始处导入了多个Python模块，包括numpy用于数学运算，functools中的reduce用于函数的累积调用，random用于生成随机数，requests用于发送HTTP请求等。

**机械臂模型建模：**机器人臂的建模基于DH参数，这是一种广泛用于机器人臂运动学的标准方法。DH参数为每一对相邻连杆定义了一个变换矩阵，包括旋转角度（theta）、连杆长度（link）、连杆偏移（d）和旋转轴（alpha）。

关键代码：

def fk(joints):

thea\_1, thea\_2, thea\_3, thea\_4, thea\_5, thea\_6 = joints

DH = [[thea\_1,d\_list[0],link\_list[0],apha\_list[0]],

[thea\_2,d\_list[1],link\_list[1],apha\_list[1]],

[thea\_3,d\_list[2],link\_list[2],apha\_list[2]],

[thea\_4,d\_list[3],link\_list[3],apha\_list[3]],

[thea\_5,d\_list[4],link\_list[4],apha\_list[4]],

[thea\_6,d\_list[5],link\_list[5],apha\_list[5]]]

**Pricefunc 类：**这个类用于计算机器人运动过程中的能耗。它初始化时接收机器人连杆的参数，如初始和结束角度、连杆长度、连杆偏移量等，并在price方法中计算运动过程中的功率消耗。

关键代码：

class Pricefunc:

def \_\_init\_\_(self,joints\_init,joints\_end,d\_list,link\_list,apha\_list,n=6,t=10,g0=9.8):

self.n=n

self.t=t

self.g0=g0

# 连杆的DH参数（Denavit-Hartenberg parameters）

# 初始和结束的关节角度

def price(self):

# 计算代价的内部函数定义和计算过程

**逆运动学（Inverse Kinematics）：**通过Inversekinematics类实现。逆运动学是计算机器人关节角度的过程，使得机器人的末端执行器能够达到一个特定的目标位置，计算出机器人各关节应达到的角度。它使用DH参数（Denavit-Hartenberg参数）来描述机器人的连杆和关节。

关键代码：

class Inversekinematics:

def \_\_init\_\_(self,joints\_init,d\_list,link\_list,apha\_list,Tar,tolerance = 1e-7):

self.DH=[[joints\_init[0],d\_list[0],link\_list[0],apha\_list[0]],

[joints\_init[1],d\_list[1],link\_list[1],apha\_list[1]],

[joints\_init[2],d\_list[2],link\_list[2],apha\_list[2]],

[joints\_init[3],d\_list[3],link\_list[3],apha\_list[3]],

[joints\_init[4],d\_list[4],link\_list[4],apha\_list[4]],

[joints\_init[5],d\_list[5],link\_list[5],apha\_list[5]]]

self.T\_tar=[[1,0,0,Tar[0]],

[0,1,0,Tar[1]],

[0,0,1,Tar[2]],

[0,0,0,1 ]]

self.tolerance = tolerance

self.joints\_init=joints\_init

def fk(self):

# 正运动学，计算当前关节角下末端执行器的位置

def rotate(axis, deg):

# 根据旋转轴和角度生成旋转矩阵

def trans(axis, dis):

# 根据移动轴和距离生成平移矩阵

T = [rotate('z', thea\_i).dot(trans('z', d\_i)).dot(trans('x', l\_i)).dot(rotate('x', a\_i))

for thea\_i, d\_i, l\_i, a\_i in self.DH]

T60 = reduce(np.dot, T)

return T60

def ik(self,link\_list):

# 逆运动学，计算达到目标位置的关节角

def fk(joints):

# 根据当前关节角计算末端执行器的位置

def get\_jac(joints: np.ndarray):

# 计算雅可比矩阵

itertime = 0

step = 0.5

joints = self.joints\_init.copy()

while itertime < 1000:

T\_cur = fk(joints)

deltaT = (self.T\_tar - T\_cur).flatten()

error = np.linalg.norm(deltaT)

if error < self.tolerance:

return joints%(2\*np.pi)

jac = get\_jac(joints)

deltaq = np.linalg.pinv(jac) @ deltaT

joints = joints + step \* deltaq

itertime += 1

return False

**FK 方法：**这是逆运动学类中的一个方法，用于执行正向运动学，即根据关节角度计算末端执行器的位置。

关键代码：

def fk(self):

def rotate(axis, deg):

AXIS = ('X', 'Y', 'Z')

axis = str(axis).upper()

if axis not in AXIS:

print(f"{axis} is unknown axis, should be one of {AXIS}")

return

rot\_x = axis == 'X'

rot\_y = axis == 'Y'

rot\_z = axis == 'Z'

rot\_mat = np.array([

[(np.cos(deg), 1)[rot\_x], (0, -np.sin(deg))[rot\_z], (0, np.sin(deg))[rot\_y], 0],

[(0, np.sin(deg))[rot\_z], (np.cos(deg), 1)[rot\_y], (0, -np.sin(deg))[rot\_x], 0],

[(0, -np.sin(deg))[rot\_y], (0, np.sin(deg))[rot\_x], (np.cos(deg), 1)[rot\_z], 0],

[0, 0, 0, 1]], dtype=np.float32)

rot\_mat = np.where(np.abs(rot\_mat) < 1e-10, 0, rot\_mat) # get a small value when np.cos(np.pi/2)

return rot\_mat

**IK 方法：**这是逆运动学类中的另一个方法，用于执行逆运动学计算，即根据末端执行器的位置计算关节角度。

关键代码：

def ik(self,link\_list):

def fk(joints):

thea\_1, thea\_2, thea\_3, thea\_4, thea\_5, thea\_6 = joints

DH = [[thea\_1,d\_list[0],link\_list[0],apha\_list[0]],

[thea\_2,d\_list[1],link\_list[1],apha\_list[1]],

[thea\_3,d\_list[2],link\_list[2],apha\_list[2]],

[thea\_4,d\_list[3],link\_list[3],apha\_list[3]],

[thea\_5,d\_list[4],link\_list[4],apha\_list[4]],

[thea\_6,d\_list[5],link\_list[5],apha\_list[5]]]

def rotate(axis, deg):

AXIS = ('X', 'Y', 'Z')

axis = str(axis).upper()

if axis not in AXIS:

print(f"{axis} is unknown axis, should be one of {AXIS}")

return

rot\_x = axis == 'X'

rot\_y = axis == 'Y'

rot\_z = axis == 'Z'

rot\_mat = np.array([

[(np.cos(deg), 1)[rot\_x], (0, -np.sin(deg))[rot\_z], (0, np.sin(deg))[rot\_y], 0],

[(0, np.sin(deg))[rot\_z], (np.cos(deg), 1)[rot\_y], (0, -np.sin(deg))[rot\_x], 0],

[(0, -np.sin(deg))[rot\_y], (0, np.sin(deg))[rot\_x], (np.cos(deg), 1)[rot\_z], 0],

[0, 0, 0, 1]], dtype=np.float32)

rot\_mat = np.where(np.abs(rot\_mat) < 1e-10, 0, rot\_mat) # get a small value when np.cos(np.pi/2)

return rot\_mat

def trans(axis, dis):

AXIS = ('X', 'Y', 'Z')

axis = str(axis).upper()

if axis not in AXIS:

print(f"{axis} is unknown axis, should be one of {AXIS}")

return

trans\_mat = np.eye(4)

trans\_mat[AXIS.index(axis), 3] = dis

return trans\_mat

T = [rotate('z', thea\_i).dot(trans('z', d\_i)).dot(trans('x', l\_i)).dot(rotate('x', a\_i))

for thea\_i, d\_i, l\_i, a\_i in DH]

T60 = reduce(np.dot, T)

return T60

**BL 方法：**这是Particles\_algorithm类中的一个方法，用于启动优化过程并返回最优的连杆长度。这个类实现了粒子群优化算法，用于寻找最优的连杆长度，使得机器人能够以最小的代价（如能耗）达到目标位置。

关键代码：

def bl(self):

best\_links = self.particle\_swarm\_optimization()

return best\_links

**粒子群优化（particle\_swarm\_optimization）：**这是Particles\_algorithm类中的一个方法，用于执行粒子群优化过程。通过Particles\_algorithm类和其内部的Particle类实现。PSO是一种基于群体智能的优化算法，它通过模拟鸟群或鱼群的社会行为来搜索最优解。

关键代码：

def particle\_swarm\_optimization(self):

iterations = self.ITERATION

swarm\_size = 10

particles = [self.Particle(self.min\_lims, self.max\_lims, self.DIMENSION,

min\_values=self.LOW\_LIMIT, max\_values=self.HIGH\_LIMIT,)

for \_ in range(swarm\_size)]

global\_best\_position = None

global\_best\_cost = float('inf')

for iteration in range(iterations):

file1 = open(r'G:\\django01\\test1.txt', 'r')

data\_str1 = file1.read()

data1 = eval(data\_str1)

p = data1.get('p')

if p == 1:

exit()

for particle in particles:

I =Inversekinematics(self.joints\_init, self.d\_list, particle.position, self.apha\_list, self.tar).ik(particle.position)

if np.isscalar(I):

current\_cost = 10\*\*7

else:

joints\_end = I

current\_cost=Pricefunc(self.joints\_init,joints\_end,self.d\_list, particle.position, self.apha\_list).price()

......

if current\_cost < particle.best\_cost:

particle.best\_cost = current\_cost

particle.best\_position = np.copy(particle.position)

if current\_cost < global\_best\_cost:

global\_best\_cost = current\_cost

global\_best\_position = np.copy(particle.position)

print(particle.position)

print('MyCost:', current\_cost)

print('BestCost:', global\_best\_cost)

for particle in particles:

inertial\_weight = 0.9 - (0.5 / iterations) \* (iteration + 1)

cognitive\_weight = 2.5 - (2.0 / iterations) \* (iteration + 1)

social\_weight = 0.5 + (2.0 / iterations) \* (iteration + 1)

particle.update\_velocity\_position(global\_best\_position,

inertial\_weight=inertial\_weight,

cognitive\_weight=cognitive\_weight, social\_weight=social\_weight)

print('One iteration finished')

return global\_best\_position

**主程序：**在\_\_main\_\_部分，程序使用示例数据初始化了粒子群优化算法，并计算了最优的连杆长度和对应的逆运动学解。

**HTTP请求：**程序中还包含了发送HTTP GET请求的代码，这可能用于与外部服务交互，如将计算结果发送到服务器。

**数学函数：**程序中定义了C和S函数，分别用于计算余弦和正弦值，以及T函数，用于构建和操作变换矩阵。

**代价函数（Cost Function）：**通过Pricefunc类实现。代价函数用于评估机器人从一个位置移动到另一个位置所需的能量消耗或其他代价。

关键代码：

class Pricefunc:

def \_\_init\_\_(self, joints\_init, joints\_end, d\_list, link\_list, apha\_list, n=6, t=10, g0=9.8):

self.n = n

self.t = t # 假设在10个单位时间完成运动，其间所有连杆匀速运动

self.g0 = g0 # 重力加速度

# 以下是连杆的DH参数和初始与结束的关节角度

# .省略了DH参数和关节角度的赋值

def price(self):

# 内部辅助函数定义

def C(x): return np.cos(x)

def S(x): return np.sin(x)

# 省略了其他内部辅助函数和计算过程

# 初始化变量

price\_sum = 0

# 通过线性插值生成一系列的关节角度

inter1 = np.linspace(self.ith1, self.eth1, 100)

inter2 = np.linspace(self.ith2, self.eth2, 100)

# .省略了其他关节的插值

# 计算每个插值角度下的代价，并求和

for \_ in range(100):

price = 0

# 遍历每个连杆，计算代价

# 省略了具体计算过程

price\_sum = price\_sum + price

return round(price\_sum, 4)

在本程序中，PSO被用于优化连杆的长度，以实现机器人臂的末端执行器达到预定目标位置。算法初始化一群粒子，每个粒子代表了连杆长度的可能解，并通过迭代过程不断更新粒子的位置和速度，以逼近全局最优解。当粒子群优化算法找到最优的连杆长度后，逆运动学（IK）被应用于计算出实现目标位置所需的关节角度。程序会计算整个运动过程的代价，包括能量消耗、时间效率和路径平滑性等因素。程序需要将计算结果通过HTTP请求发送到机械臂模型页面。程序将计算得到的最优连杆长度和关节角度写入到文本文件中，以便于后续的分析、记录或作为其他程序的输入。

## 5.5本章小结

为了使整个系统具有绿色可持续性，本章主要介绍了正向运动学分析、逆向运动学分析和改进型粒子群算法。来解决机器人学的逆运动学问题，通过算法找到最优的连杆（link）长度，本文提出了一种基于粒子群优化算法（PSO）的方法，用于解决机器人臂的逆运动学问题，并计算其运动过程中的动力学成本。通过定义机器人臂的DH（Denavit-Hartenberg）参数，利用正运动学（FK）和逆运动学（IK）求解器，结合动力学分析，优化机器人臂的运动路径，以最小化能量消耗，使得六轴机器人臂能够以最小的代价（例如能耗或成本）到达指定的目标位置。

设置目标点为[0.4, 0.7, 0.3]，点击机械臂编辑页面的启动算法按钮启动算法，得到的结果如图5. 2所示，并将数据储存在aaa.txt文件中以便查看，每次迭代的数据都会发送到机械臂模型页面，让机械臂模型动态显示，以便于观察每次迭代的结果。

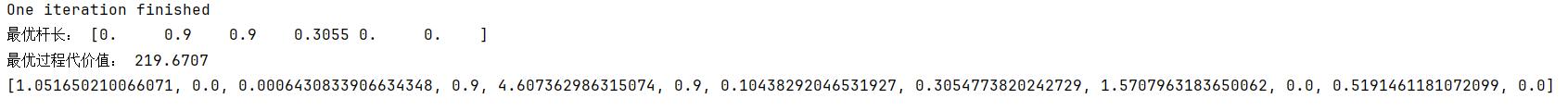


图5. 2 粒子群算法迭代结果

# 第6章 总结与展望

可编辑调试的机械臂数字孪生可视化系统构建对于机械臂数字孪生具有重大意义。在 Jadon等人[7]的研究中介绍了在Django框架下开发Web应用的方法和Martínez等人[18]在研究中讨论了基于Three.js的设备控制系统的监控屏应用和发展前景等方法的启发下，通过后端Django技术、前端Three.js技术、数据库Redis技术和WebSocket通讯技术来实现可编辑调试的机械臂数字孪生可视化系统构建。本课题的实现对于智能制造与自动化、数据处理与存储、系统可编辑调试性以及科研与教学等方面都具有重大意义。通过深入研究和实践，可以推动相关技术的发展和应用，为工业4.0和智能制造领域的发展做出贡献。

为了使机械臂数字模型能够动态实时的展示给用户，前端采用Three.js技术。在构建可编辑调试的机械臂数字孪生系统中，前端Three.js技术主要研究工作集中在如何利用Three.js创建的机械臂3D模型，并实现其在网页端的实时渲染和交互。Three.js提供了丰富的3D图形编程接口，能够灵活地构建复杂的3D场景，包括机械臂的各个关节、连杆以及其他辅助设备。深入研究Three.js的渲染机制、光照模型以及材质贴图等关键技术，以确保生成的机械臂模型既真实又具备高度的可视化效果。还利用Three.js的动画和交互功能，实现对机械臂运动轨迹的模拟和用户操作的实时响应，使用户能够直观地理解和操作数字孪生系统中的机械臂。这些研究内容不仅提升了系统的用户体验，也为后续的数字孪生技术应用提供了有力的技术支撑。

由于此系统涵盖用户注册登录、机械臂编辑页面和机械臂模型页面，会产生大量的数据需要处理，为了使数据快速处理并实时传输，后端采用Django技术。Django技术作为可编辑调试的机械臂数字孪生系统的核心框架，主要负责处理系统的业务逻辑、数据交互以及提供用户交互接口。通过Django的模型层（Models），可以定义并操作与机械臂数字孪生相关的数据结构，如用户信息、机械臂状态数据等，确保数据的完整性和一致性。视图层（Views）则负责处理用户请求，通过调用业务逻辑代码，将用户请求转化为数据库操作或生成前端页面所需的HTML响应。同时，Django的模板系统可以方便地创建和修改用户交互界面，实现用户注册、登录等功能。此外，Django还提供了强大的URL路由系统、表单处理、中间件等功能，使得后端开发更加高效、灵活，从而支撑起整个机械臂数字孪生系统的稳定运行和持续优化。

为了避免用户账号数据丢失以及登陆时账户与密码快速匹配，本课题采用Redis技术作为系统的数据库。数据库Redis技术主要负责用户注册登录信息的快速存储与访问。Redis以其高速的内存存储、灵活的数据结构以及丰富的功能API，为系统提供了高效的用户认证与会话管理机制。在系统中，Redis被用作缓存层，用于存储用户登录凭证（如会话令牌、用户ID等），这些信息在用户登录时被创建并存储于Redis中，确保用户在整个会话期间能够快速验证身份并访问系统资源。此外，Redis还支持数据持久化，确保在系统重启或故障时，用户登录信息不会丢失。通过Redis，我们能够实现用户登录的快速响应、高并发处理以及系统安全性的提升，为可编辑调试的机械臂数字孪生系统提供稳定可靠的用户认证服务。

可编辑调试的机械臂数字孪生可视化系统构建工作框架如图6. 1所示。

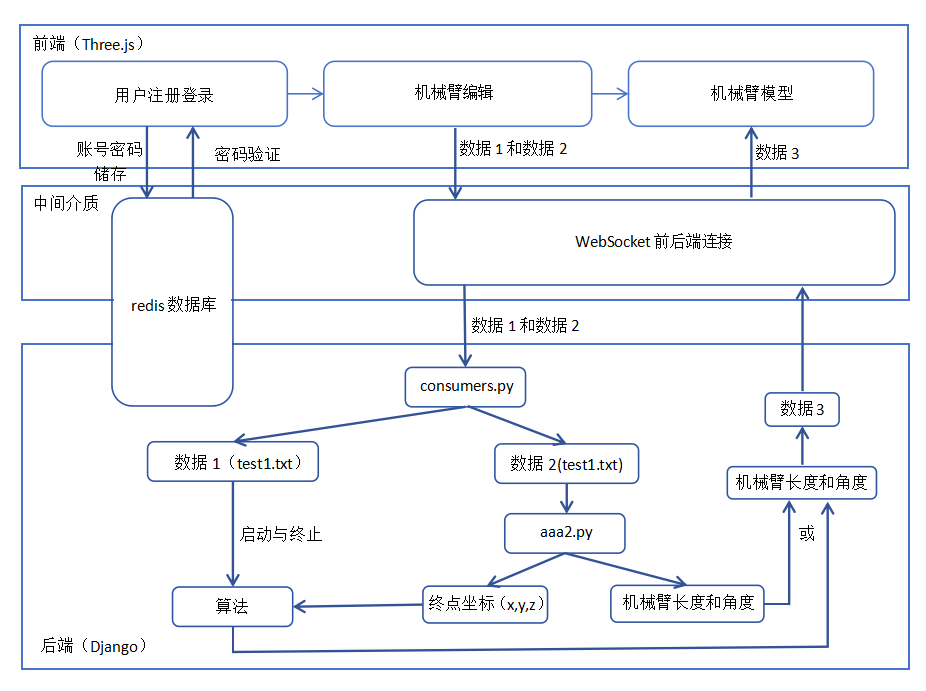


图6. 1 可编辑调试的机械臂数字孪生可视化系统构建框架

随着技术的进步和工业互联网的深入发展，该系统的应用场景将更加广泛，不仅仅局限于传统的工业生产线，还有可能拓展到医疗、航空航天等高精度、高要求领域。

Django后端技术的强大功能和灵活性，使得系统能够轻松应对复杂的数据处理和业务逻辑，为数字孪生系统提供了稳定可靠的后端支持。前端Three.js技术的应用将进一步提升系统的交互性和可视化效果，使用户能够更加直观地观察和控制机械臂的运动状态。

Redis数据库的高效性能和快速响应能力将确保用户注册登录等信息的实时处理，提升用户体验。而WebSocket通讯技术的应用，将实现数据的实时传输和同步，使得数字孪生系统能够实时反馈机械臂的状态，便于远程监控和调试。

随着系统的不断完善和优化，未来这一技术希望可以在工业自动化领域提供一定借鉴。机械臂数字孪生系统不仅可以提高生产效率和降低运维成本，还能通过精确的数据分析帮助企业做出更科学的决策。

可编辑调试的机械臂数字孪生可视化系统构建课题具有深远的意义和广阔的应用前景。不仅体现了信息技术与工业技术的深度融合，而且预示着未来智能制造和工业自动化的发展方向。

# 参考文献

1. Grieves M. Digital twin: manufacturing excellence through virtual factory replication[J]. White paper, 2014, 1(2014): 1-7.
2. 陈玉娇,曾诗雨,红杰，等。工业机器人码垛数字孪生系统的研究与实现[J].计算机集成制造系统,2023,29(06):1930-1940.
3. 陈述,鲁世立,王建平，等。融合Unity3D的缆索起重机安全运行数字孪生模型构建方法[J].中国安全生产科学技术,2024,20(01):154-159.
4. 周高伟,沙杰,刘梦园,等.基于数字孪生的加工生产线虚实交互技术研究[J].机电工程,2024,41(02):337-344.
5. 杨思楠.机械臂3D陶土打印技术研究及其数字孪生工厂构建[D].石家庄铁道大学,2023.
6. 林润泽,王行健,冯毅萍，等.基于数字孪生的智能装配机械臂实验系统[J].实验室研究与探索,2019,38(12):83-88.
7. Thakur and P. Jadon, "Django: Developing web using Python," *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Greater Noida, India, 2023, pp. 303-306, Yu、Xiyaetal.“基于Django的住房信息管理系统的设计和部署”。JournalofPhysics：会议系列2425（2023）。
8. D. Martinez, D. Gonzalez, J. M. Sánchez and R. M. Toasa, "Library in django framework to standardize early-stage web application development," *2023 18th Iberian Conference on Information Systems and Technologies (CISTI)*, Aveiro, Portugal, 2023, pp. 1-4.
9. 黄平,戈新生(2017).基于能量法的空间机械臂的姿态控制稳定性。国际航空航天科学, 2017, 5(1), 45‐53
10. 陈伟,李敬兆,石晴,等.多目标优化的智能胶带输送系统数字孪生模型构建及其应用研究[J/OL].煤炭科学技术，2024-04-22.
11. 周翔,贺兴,陈赟,等.超大型城市虚拟电厂的数字孪生框架设计及实践[J/OL].电网技术，2024-04-22.
12. 李颖,熊冰,桑笑晗,等.沉浸式力学实验数字孪生系统架构研究[J/OL].工程力学:1-13[2024-04-22]
13. 韩冬阳,夏唐斌,范宜静等.智能制造系统的数字孪生正向监测与反向控制方法[J/OL].计算机集成制造系统:1-23[2024-03-12].
14. 曹宇,甘霖,张涛.基于数字孪生的海上浮式发电平台连接器应力场预报方法[J/OL].中国舰船研究:1-11[2024-03-12].
15. 温琼阳,朱学军,李毅,等.基于改进粒子群算法的机械臂能耗轨迹优化[J/OL].计算机应用研究:1-8[2024-05-02].
16. 吴庆宗,胡兴柳,周智慧.基于改进粒子群算法的六自由度机械臂轨迹优化[J].盐城工学院学报(自然科学版),2023,36(04):41-47.
17. 赵亮,刘瑞雪,张玮奇,等.基于自适应混沌麻雀搜索算法的机械臂最优时间轨迹规划[J/OL].信息与控制:1-11[2024-05-02]
18. D. Martinez, D. Gonzalez, J. M. Sánchez and R. M. Toasa, "Library in django framework to standardize early-stage web application development," 2023 18th Iberian Conference on Information Systems and Technologies (CISTI), Aveiro, Portugal, 2023.
19. 牟奇杰,项阳,罗泽鑫,等.基于数字孪生的机械臂[J].物联网技术,2024,14(04):123-126.
20. 杨洪.基于Redis的共享缓存分析[J].数字技术与应用,2024,42(01):110-112.

# 致 谢

人生天地之间，若白驹过隙，忽然而已。行文至此，意味着大学四年即将结束，也意味着学生时代即将结束。驻足此刻，任由记忆的闸门缓缓打开，那些往昔的点滴如潮水般涌来，让人不觉的沉溺与此。

寸草春晖，难以为报。感谢国家对我们贫困生的照顾，学生没有大作为，感谢只能表达在字里行间，但学生不会停止学习的脚步，只为中华之复兴而读书；感谢南昌航空大学，营造了一个既温馨又富有青春活力的学习氛围，感谢所有老师，尤其是张冉阳老师，平时教学的仔细和生活的关心，给我的大学生添加了一份曙光；感谢我的家人，是他们始终给予我坚定的支持和无尽的关爱。在我迷茫、疲惫的时候，家人总是我最坚实的后盾，他们的鼓励和陪伴让我勇往直前，无惧任何挑战。我深知，有家的温暖和支持，我才能更加勇敢地面对生活中的一切。

愿岁并谢，与友长兮。感谢一起生活的三个室友，陪我度过无数快乐的时光，正是因为有了你们，我的大学才变得多姿多彩。追风赶月莫停留，平芜尽处是春山，天下没有不散的宴席，但人生何处不相逢。愿你们此去前途似锦，归来仍是少年，此去一别，奔赴山海，顶峰相见。

道阻且长，行则降至。都说大学都是充满青春的，而我的大学却非如此，充满的是三点一线的生活，没有太多的色彩，总觉得等自己变得优秀再去做喜欢的事、买喜欢的东西、追求喜欢的人，可是总事与愿违。没等到变得优秀，却被一次次的失败抹平了棱角，也好像接受了自己不优秀的一面，认为没有资格去做喜欢的事和追求喜欢的人。又好像只有优秀的人才配提起过往的辛酸。但我相信“苦尽甘来总有时，一路向阳待花期”，我终究会成为心中所向的那个人。

愿母校滋兰树蕙，永续化意！愿祖国繁荣昌盛，山河无恙！愿家人朋友平安健康，幸福美满。愿自己不忘初心，砥砺前行。