2022 年英特尔杯大学生电子设计竞赛嵌入式系统专题邀请赛

2022 Intel Cup Undergraduate Electronic Design Contest

- Embedded System Design Invitational Contest

作品设计报告

Final Report

Intel Cup Embedded System Design Contest

报告题目：基于 RGB-D 相机的视觉牵引机械辅助系统

学生姓名：冯玉泽、唐建凯、朱墨

指导教师：佘黎煌

参赛学校：东北大学

2022 年英特尔杯大学生电子设计竞赛嵌入式系统专题邀请赛

参赛作品原创性声明

本人郑重声明：所呈交的参赛作品报告，是本人和队友独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果，不侵犯任何第三方的知识产权或其他权利。本人完全意识到本声明的法律结果由本人承担。

参赛队员签名：

日期：　　2022 年　7 月　25 日

基于 RGB-D 相机的视觉牵引机械辅助系统

摘要

本项目是基于深度学习的 RGB-D 视觉牵引机械辅助系统，由 Intel 边缘 AI 计算平台和搭载仿生机械手臂的移动作业平台组成，普适于远程无人作业。使用者可单人通过踏板远程移动平台至作业区，通过 SLAM 与可动相机得到位置反馈，利用 RGB-D 相机实时提取手部信息，优化后投入 PointNet++神经网络计算，得到 21 关键节点，通过仿生手臂运动算法计算出各舵机旋转角度，通过 Wi-Fi 或 4G/5G 将信息传输至仿生手臂控制平台，做出相应的姿势，完成远程实时牵引控制。

关键词：深度相机，PointNet++网络，手势识别，仿生机械手臂运动算法，实时视觉控制机械手臂，边缘计算

VISUAL TRACTION MECHANICAL ASSISTANCE SYSTEM BASED ON RGB-D CAMERA

ABSTRACT

This project is an RGB-D visual traction mechanical assistance system based on deep learning, which consists of Intel edge AI computing platform, visual recognition terminal, bionic hand control terminal, and mobile platform control terminal. It is suitable for remote unmanned universal operations. The user can remotely control the mobile platform to the work area through the pedal, obtain position feedback through the slam and movable camera, extract the hand information in real time by using the RGB-D camera, and put it into the PointNet++ neural network calculation after optimization, and obtain 21 key nodes. The bionic hand motion algorithm calculates the motion angle of each steering gear, transmits the information to the bionic hand control module through Wi-Fi or 4G/5G module, makes corresponding postures, and completes real-time mechanical traction.

Key words: RGB-D camera, PointNet++, bionic hand motion algorithm, real-time mechanical traction, edge computing

目 录

第一章 绪论 1
1.1 系统的研究意义与背景 1
1.2 国内外研究现状 1
1.2.1 手势识别技术 1
1.2.2 仿生手控制技术 2
1.3 主要研究内容及组织结构 2
第二章 系统总体设计方案 3
2.1 系统总体设计方案概述 3
2.2 方案可行性分析 4
第三章 手势识别处理模块设计 5
3.1 RGB-D 相机校准与采集参数修改 5
3.2 点云特性 6
3.3 PointNet++ 模型 7
3.3.1 PointNet 网络结构 7
3.3.2 PointNet 手势估计网络 8
3.3.3 数据集 8
3.3.4 PointNet 模型训练 9
3.4 深度图像转换点云处理 10
3.5 PointNet 手势关键点识别 11
3.5.1 OBB 转换 11
3.5.2 PointNet++三维手势估计网络结构 12
第四章 机械手臂运动控制模块设计 14
4.1 机械手模型与运动算法设计 14
4.1.1 机械手结构原理 14
4.1.2 机械手掌运动算法的设计 15
4.2 机械臂模型与运动算法设计 18
4.2.1 机械臂模型建立 18
4.2.2 机械臂运动算法设计 19

4.3 仿生手臂控制系统硬件设计 21
第五章 移动平台的 SLAM 建模与远程控制模块设计 23
5.1 基于 STM32F103 的移动平台远程控制设计与实现 23
5.2 基于 ROS 的激光雷达 SLAM 建模 24
第六章 系统测试 25
6.1 RGB-D 手势识别算法测试 25
6.2 仿生机械手控制测试 27
6.3 系统测试 28
第七章 关键技术及创新点 29
7.1 系统功能实现 29
7.2 系统特色与创新点 29
7.3 系统关键技术 29
附录 32

第一章 绪论

1.1 系统的研究意义与背景

仿生手作为应用广泛的自动化装备，已经有 50 多年的发展历史，凭借着其高效率，高精度和稳定性好等优点，已经在各种高危环境工作中逐步替代人工进行作业，在智能制造，医疗手术中也发挥着重要作用[1]。

目前，大多数仿生手都是根据特定工作任务和固定的操作环境进行设计，虽然可以保证较好的稳定性，但是也存在着灵活性和适应性差的缺点。传统的人机交互方式操作复杂，实时性较差，而各类数据手套虽然精度较高，但是只能用于特殊场景，适应性也较差，对仿生手的控制有着诸多的限制，难以大范围应用推广。

手作为仿生手模拟的对象，手势也是最直观的表达方式，更加自然，具有很强的视觉效果，能直接实时反映出对仿生手的控制效果。通过摄像机直接采集用户手势图像并进行识别，不仅降低了应用成本，同时通过手势关键点可直接对应控制仿生手各节点，降低了通过其他方式控制需要进行复杂的计算和转换过程。

1.2 国内外研究现状

1.2.1 手势识别技术

手势通过不同的动作、组合等，能够非常直观表达和传递丰富的信息，在很早就成为各国的研究热点，近年来，随着人机交互的高速发展，手势识别与自动驾驶、虚拟现实、机器人等各种新领域交叉应用也越来越深入。

手势识别主要分为通过可穿戴数据手套采集数据用于识别和相机获取图像进行识别。

基于传感器手套采集生物信息进行手势识别，必须要在手上戴上复杂的手套，随着 MEMS 技术的发展，用于数据手套上的传感器做到更加微型化、智能化，通过对手势建模和分析，更好的进行手势估计[2]。近年来，对于生物电信号的研究逐渐成为热点，通过获取手部肌肉信号的变化来识别手势[3]。

基于视觉图像的方法一般通过一个或者多个摄像头采集手势图像，然后通过分割算法获取图像中的目标手势。与可穿戴设备相比，使用者无需佩戴额外的装置，便可以获取精度较好，效果更直观的手势数据。

在深度摄像头出现之前，对于手势的识别还是基于传统的 2D 图像，通过对图片信息进行预处理，手势分割提取，完成的手势的分析和识别。预处理首先把图片中的手定位出来，并通过分割把手势区域与背景分割出来，然后通过视觉特征检测或者多模式融合检测等方法识别出手势[4]。

基于 RGBD 的手势识别方法主要分为基于预先构建的 3D 手部模型的方法、基于手势表面特征判别的方法和将两种方法相结合进行综合判别的方法。

模型方法需要通过深度学习方法，首先将手部数据拟合成数学模型，与手势数据集训练出的手部模型进行手势识别[5]。由于人手的模型比较复杂，通常在最初训练是，选用的都是简单几何体的组合，能很好的简化训练任务。2015 年，浙江大学 Peiyi Li 等提出的随机决策森林框架 3D 手势估计算法，通过深度图像数据输入，得到手部的关节[6]。卷积神经网络的发展，也为手势识别提供了一种新方案。2017 年，Liuhao Ge 等使用 3D CNN 方

法，通过把深度图像数据输入 3D CNN 网络中，得到手势的关键点信息[7]。2018 年，美国斯坦福大学的 Liuhao Ge 等提出的基于 PointNet++的点云手势估计网络，通过深度图像生成的点云作为输入，采用端到端网络，从点云中识别出手部姿势[8]。

基于深度数据手势识别一般包括使用深度相机采集深度数据，对人手进行建模，数据标注，人手定位与手势估计，如图 1-1 所示。

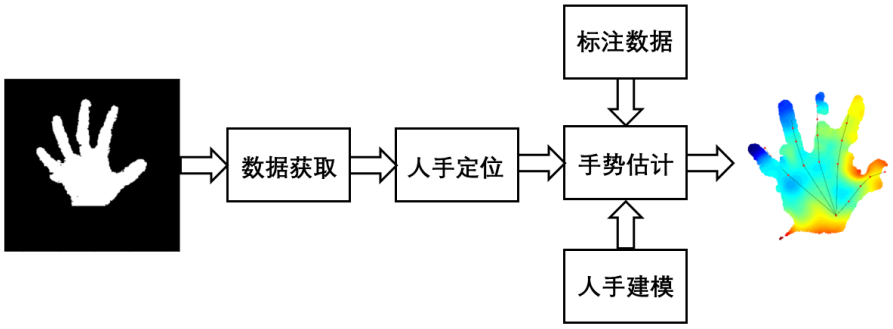


图 1-1 手势估计流程

1.2.2 仿生手控制技术

进入 21 世纪，仿生手的相关技术有了很大的进步。如 Shadow 公司研发的 24 自由度的 Shadow 手，具有很高的灵敏度。由哈尔滨工业大学开发的仿生手 HIT/DLR II，采用模块化结构，15 个自由度，其传动机构为齿轮和带传动，将电机、减速机构、传感器、驱动器高度集成在人手大小中，工艺及其复杂，灵巧手具备力矩、指尖力等传感器反馈[9]。日本 Gifu 大学研制出的用于医疗领域的五指仿生机械手 Gifu-II[10]。采用电机驱动，同时还装备了触觉传感器和六轴向力传感器，具有很高的灵巧性[11]。

1.3 主要研究内容及组织结构

本系统为基于 RGB-D 相机的视觉牵引机械辅助系统，共分成三个具体模块，分别是手势识别模块、仿生手运动控制模块、远程移动平台模块。

本文各章节的内容如下：

第 1 章阐述了进行深度图像手势识别与仿生手控制的研究背景意义与研究现状。

第 2 章介绍了系统总体设计方案与流程。

第 3 章主要介绍了手势识别模块，通过改进 PointNet 网络进行手势估计。解释整个网络的运行步骤，详细阐述算法的具体实现。

第 4 章主要介绍了整个控制系统的搭建与控制工作，着重介绍仿生手控制算法与仿生手运动算法模型。

第 5 章介绍了移动控制平台的 slam 建模功能与远程控制模块的设计理念。

第 6 章总结了整个系统的系统级测试与最终效果展示。

第 7 章阐述整个项目的关键技术及创新点。

第二章 系统总体设计方案

2.1 系统总体设计方案概述

本系统是基于深度学习构建的 RGB-D 视觉牵引机械辅助系统，由边缘计算平台和作业平台构成，如图 2-1。

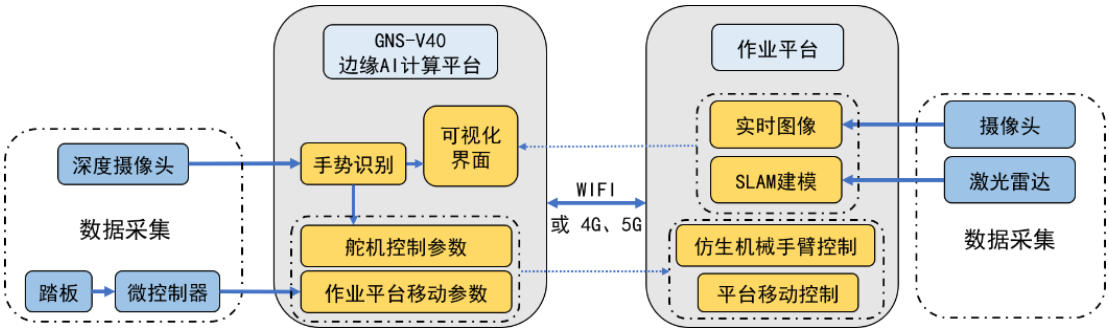


图 2-1 视觉牵引机械辅助系统模块化架构图

本系统使用 RGB-D 深度相机采集手部信息，传输至 Intel 边缘计算平台，采用 Open3D 工具将深度图像转化为点云。配准优化与 OBB 归一化后，再将点云投入训练好的 PointNet 深度神经网络中，进行手势姿态估计，计算出 21 个关键节点的 3 维位置信息，通过机械手臂运动算法，计算出各舵机的运动角度。通过网络将数据送至作业平台，完成对仿生机械手臂控制。使用者也可以使用踏板远程操作移动平台，搭载机械臂前往作业区域。

系统使用的硬件主要由 Intel 边缘计算平台、接收踏板信息的微控制器、操控仿生机械手臂运动以及作业平台移动的控制平台组成，系统硬件框图和程序流程图如下：

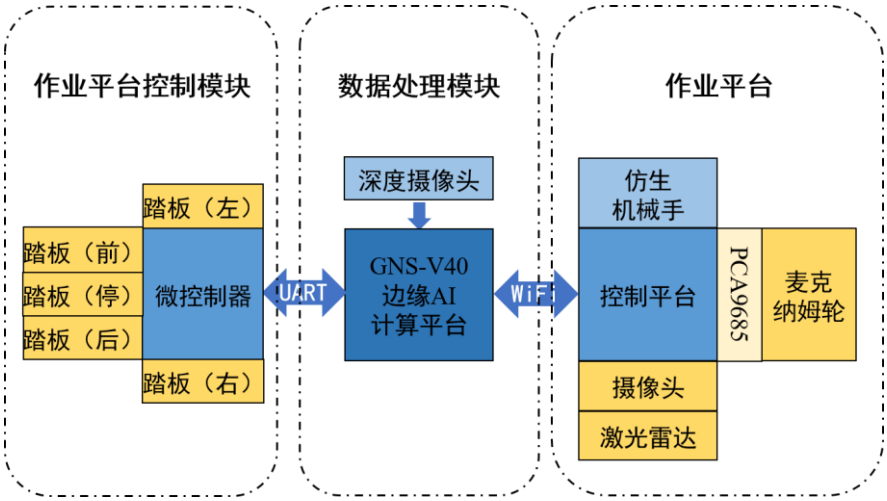


图 2-2 系统硬件框图

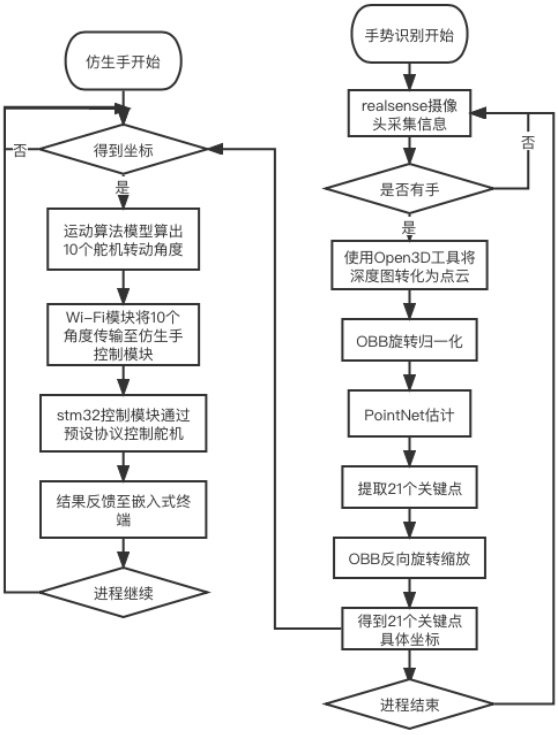


图 2-3 程序流程图

2.2 方案可行性分析

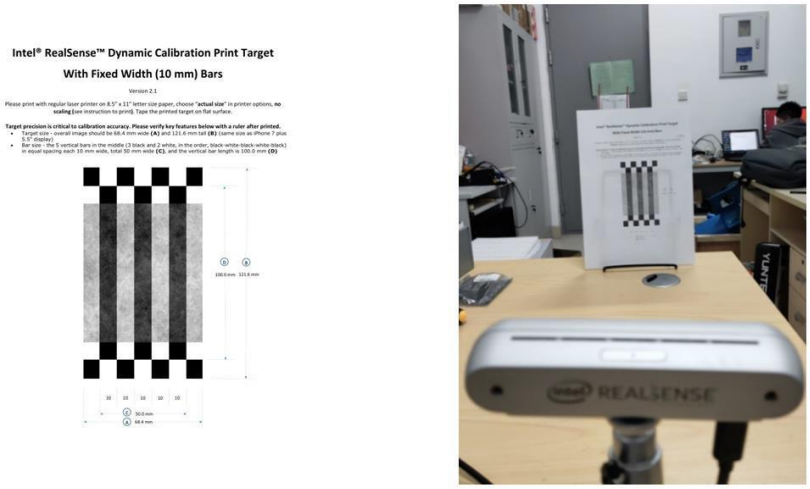
系统所用平台为信步 GNS-V40 边缘 AI 计算平台，搭载 Intel i5-1135G7 4 核处理器。高达 4.2GHz 的主频与 Intel Iris Xe Graphics 集成显卡的加成，完全地满足了整个系统所有的处理器需求。同时，平台提供的丰富接口与多张网卡，保证了本项目系统中的大量远程无线通信与各模块之间互相交互的需求得到满足。六个 USB3.2 高速接口支持 16384 Byte 块传输，使得深度摄像头采集的深度图像信息能确保完整快速传输。

得益于边缘 AI 计算平台的强大性能，整个系统得到了最大的性能释放，传统系统短板的传输模块与通信模块也能借助平台性能达到较高水平。平台在大小与重量方面也有着巨大的优势，使得系统兼具便携性、易达性。

第三章 手势识别处理模块设计

3.1 RGB-D 相机校准与采集参数修改

手势识别通常将 RGB 图像和 RGBD 图像作为数据输入，利用手势估计网络从输入图像中估计出手势的关节位置，在仿生手控制领域、虚拟现实和自动驾驶等领域也有着广泛的应用[12]。手势识别研究中早期主要使用 RGB 图像，由于 RGB 图像只是平面的二维表示，不包含 3D 场景信息，识别出的手势准确性较低。随着传感器的不断进步，英特尔 Realsense RGBD 深度相机强大的性能用于采集三维场景深度信息，为手势识别提供了很好的硬件支持。



(a) 校准目标　　(b) 校准阶段

图 3-1 动态校准过程

RealSense 动态校准（Dynamic Calibration）适用于 RealSense 摄像头校准，打开校准工具将摄像头距离目标距离在 60-85cm，推荐在 70cm 左右最后好。

Intel Realsense D455 可以采集 RGB 图像和深度图像。为了确保可以实时对应深度图像与 RGB 图像手势关节点信息，深度相机采集的深度图像通常需要配准。设 RGB 图像的像素为$(u_R, v_R, z_R)^T$，深度图像的像素为$(u_L, v_L, z_L)^T$，变换矩阵W'：

$$\begin{bmatrix} u_R \\ v_R \\ z_R \end{bmatrix} = W' \cdot \begin{bmatrix} u_L \\ v_L \\ z_L \end{bmatrix} \tag{3-1}$$

$$z_L \cdot \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{dx} & 0 & u_L^0 & 0 \\ 0 & \frac{f}{dy} & v_L^0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_L \\ y_L \\ z_L \\ 1 \end{bmatrix} \tag{3-2}$$

$$\begin{bmatrix} u_R \\ v_R \\ 1 \\ \frac{1}{z_R} \end{bmatrix} = W \cdot \begin{bmatrix} u_L \\ v_L \\ 1 \\ \frac{1}{z_L} \end{bmatrix} \quad (3-3)$$

3.2 点云特性

2017 年 Charles R. Qi 等提出了点云处理的网络 PointNet，该网络省去了将点云转化为其他形式数据的过程，将点云不经过转换，作为数据输入，在物体分类、区域分割和场景语义分析等都取得了广泛的应用，这也是本项目用到的方法。

三维数据包含位置信息，能更好的反映物理世界中事物和运动的详细情况。本项目中，三维数据选择用点云进行表示。

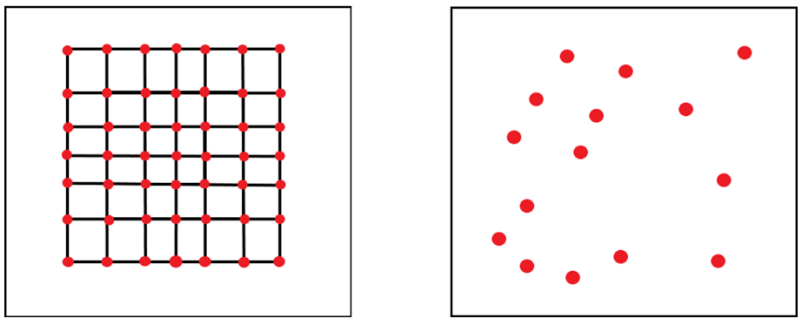


图 3-2 图像和点云

点云（Point-Cloud）能描述每个点的信息。如图 3-2 所示，和图像相比，图像的数据分布更像是一个矩阵，而点云的数据分布更像是一对点的集合，点云是无序、稀疏、不规则分布的，点云中点的坐标可以是任意值。

相较于传统的二维图像，点云的深度信息使得它包含了丰富的空间信息，从而能够表现出物体相关信息。一般来讲，由点云构成的点集主要有 3 个特点：

（1）点云无序性（Unordered）。与图像的像素点规则的矩阵排列不同，点云是无序的点。点云是一个三维空间坐标的集合。当对这 N 个点输入时，共有 N 种不同组合，如图 3-3 所示。

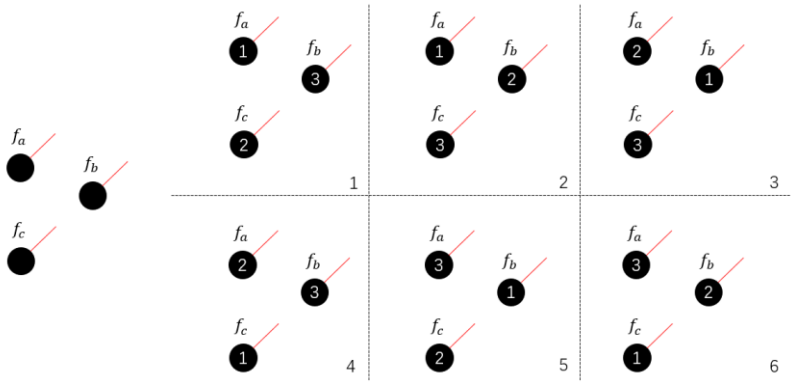


图 3-3 点云的无序性

（2）点云的相互作用（Interaction among points）。点云不是孤立的，相邻点形成一个

子集，局部点云之间的组合可以相互作用。

（3）变换下的不变性（Invariance under transformations）。点云是一个三维空间中物体表面特征的集合，不会因为点云的移动或旋转对物体本身的性质产和形状产生改变，所以它所表示的物体的特征在经过这种变换后也依然保持不变，如图 3-4 所示[13]。

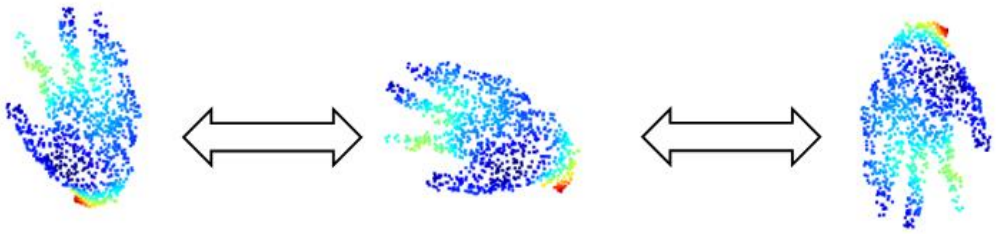


图 3-4 变换下的不变性

3.3 PointNet++ 模型

训练和测试所用数据集都来自开源库下载。

3.3.1 PointNet 网络结构

由于点云是无序、稀疏、不规则分布的，因此不能相处理传统图像的方法来处理点云数据。为了能从点云集合中得到深度信息， Charles R. Qi 等人在 2017 年提出的 PointNet 网络模型。其网络结构如图 3-5 所示。PointNet 网络的整体结构分为分类网络(Classification Network)分割网络（Segmentation Network）两部分。

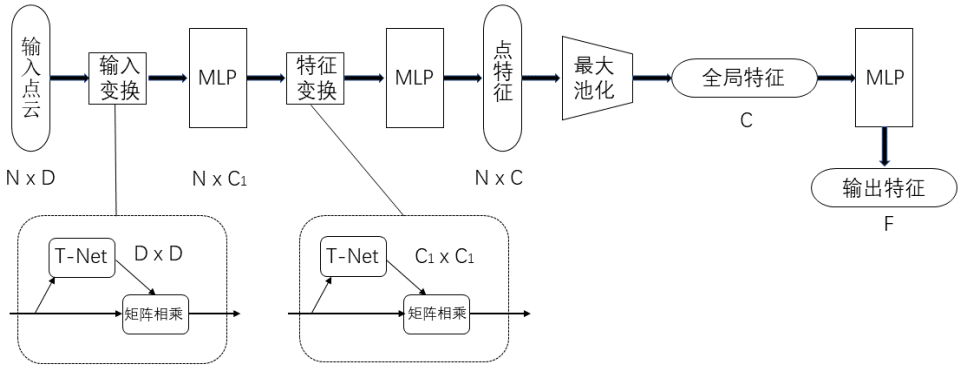


图 3-5 PointNet 网络结构

由于点云具有无序性、点间相互作用性、旋转不变性。针对点云的特征，PointNet 采用以下的技术处理点云数据：

（1）最大池化操作：经过共享权重的 MLP 以及两次特征提取后，通过最大池化操作，从而做到特征压缩融合，获得与点云次序无关的全局向量。

（2）共享权重的 MLP：在共享权重的 MLP（shared MLP）中，输入为包含多点的点云，对其中的每一个点乘相同的权重，即为共享权重。该方法对将点云中每个点视为集合中同性质的元素，对所有的点进行相同的特征提取操作，也能够解决点云的无序性问题。

（3）点云特征与点特征的融合：在 PointNet 中，先将经过最大池化后得到的全局特征向量进行了维度上的扩充，扩充为 $n \times 1024$ 的矩阵。然后将扩充后的矩阵与第 3 步得到的 $n \times 64$ 维度的特征矩阵拼接，经过共享权重的 MLP，融合了点云的全局特征和点云每个点的局部特征。

3.3.2 PointNet 手势估计网络

PointNet 手势估计网络用于 RGBD 图像中的 3D 手姿势估计，如图 3-6 所示。将深度转换为点云，并下采样为 N 个点，通过 OBB 对手的 3D 点云在定向边界框中归一化，以使的识别的方法适合各种手动方向。PointNet 网络将输入点云在估计的表面输入的归一化点的 3D 坐标中，并输出 3D 手关节位置的低维表示，然后在相机坐标系统中恢复手势关键点信息。

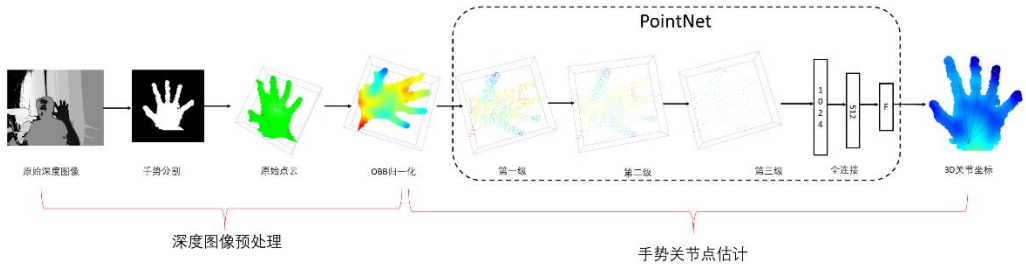


图 3-6 网络整体结构

从图 3-6 中可以看出，在方向的边界框中使用 3D 点云实现了数据标准化和旋转（OBB）和局部特征提取来方法来提取手部关节特征。在采样过程中，点云数据依次降低三个级别，其中 1 级数据标准化为 1024 点；2 级包括 512 点；3 级包括 128 点。

局部特征提取方式又可分为多尺度成组 (MSG)和多分辨率成组 (MRG)。

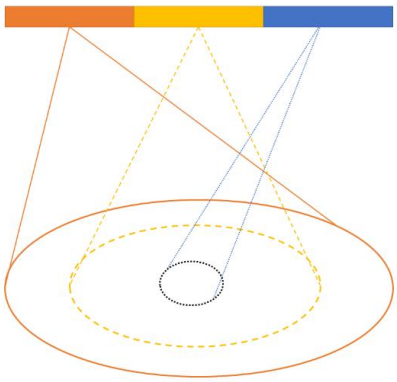


图 3-7 局部特征提取

多尺度成组(MSG)如图 3-7 是从点云中选择一个靠近中心的点云作为中心点，以该点为圆心选择多个不同的半径作为各组数据，并通过 PointNet 对不同半径的组进行抽取特征，并把不同组的特征进行组合作为该点的特征。

3.3.3 数据集

目前开源的手势数据集有很多，其中用于深度图像的数据集主要有：ICVL、NYU、MSRA15[14] [15] [16]。经过对比，我们最终使用的是 MSRA15 数据集。

MSRA15 数据集是深度图像手势领域最复杂的数据集。训练集包含标记数据为 21 个关节，有 9 个人的手势数据，17 个手势，共 76375 帧，它以迭代的方式进行注释，通过使用优化方法和手动重新调整交替进行，直到收敛。手从图像中分割出来，并且真值(Ground

Truth)包含 M = 21 个关节。训练时对所有受试者重复该过程 9 次并得到平均指标。

表 3-1 手势数据集

数据集	图像数量	类别数	关节数
ICVL	17604	10	16
NYU	81009	2	36
MSRA15	76375	9	21

3.3.4 PointNet 模型训练

对于训练 PointNet 模型，初始学习率为 0.001，batch size 为 96，正则化强度(regularization strength)为 0.0005[17]。在 50 次训练学习后，学习率除以 10。训练过程在 60 次 epoch 后停止。对于指尖细化，我们将最近邻点数 K 设置为 256。考虑到实时性，当数据集中的点云中的点数超过上限 N'= 6000，将 Kd-tree 算法应用于有效的最近邻搜索[18]。详细的参数设置如下图 3-8 所示。

```
parser = argparse.ArgumentParser()
parser.add_argument('--batchSize', type=int, default=96, help='input batch size')
parser.add_argument('--workers', type=int, default=0, help='number of data loading workers')
parser.add_argument('--nepoch', type=int, default=90, help='number of epochs to train for')
parser.add_argument('--ngpu', type=int, default=1, help='# GPUs')
parser.add_argument('--main_gpu', type=int, default=0, help='main GPU id')
parser.add_argument('--learning_rate', type=float, default=0.001, help='learning rate at t=0')
parser.add_argument('--momentum', type=float, default=0.9, help='momentum (SGD only)')
parser.add_argument('--weight_decay', type=float, default=0.0005, help='weight decay (SGD only)')
parser.add_argument('--learning_rate_decay', type=float, default=1e-7, help='learning rate decay')
```

图 3-8 PointNet 训练参数

经过 60 次 epoch，模型的每次训练迭代的单个样本的均方误差与测试均方误差如图 3-9（左）所示，模型的在世界坐标系下的训练平均估计误差与测试平均估计误差如图 3-9（右）所示。

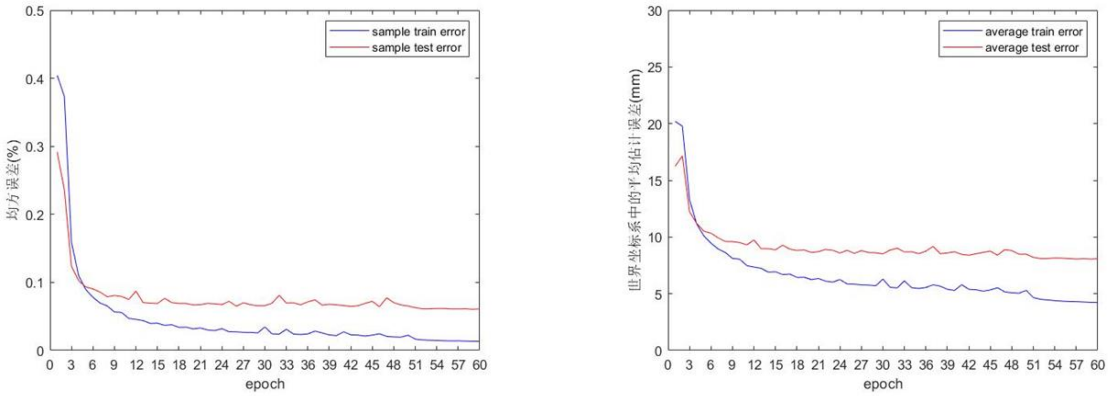


图 3-9 训练均方误差

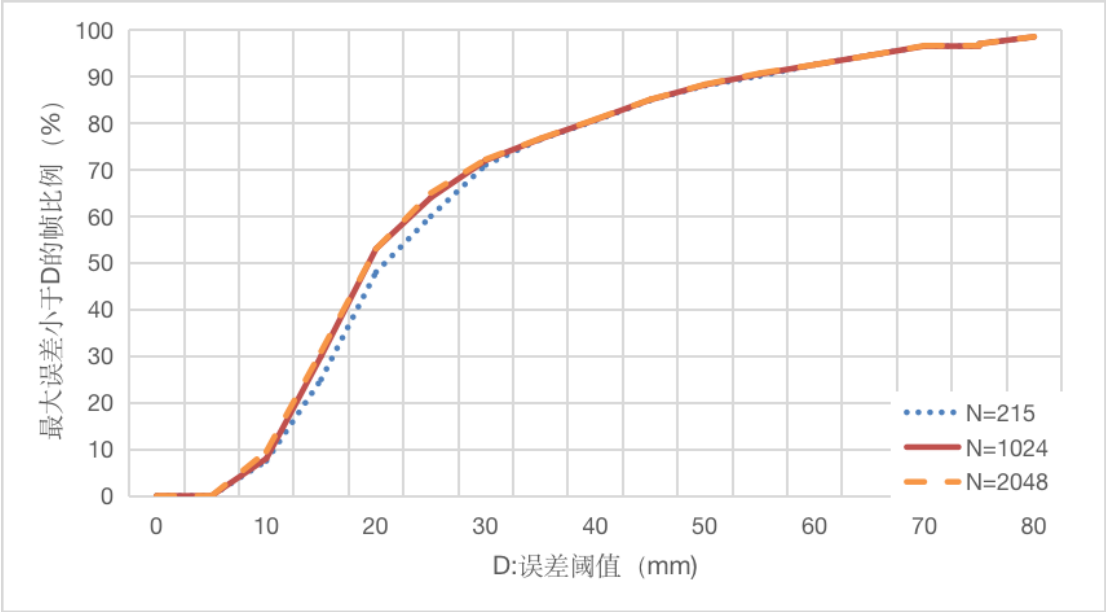


图 3-10 不同采样点数对错误帧比例的影响

为了评估采样点数 N 的影响。如图 3-10 所示，通过使用基本 PointNet 对 MRSA 15 数据集进行手部姿势回归来对三种不同数量的采样点进行实验。当 N = 512 时，估计精度略低于其他两个采样点数较多的结果。但是当 N = 1024 和 N = 2048 时，估计精度几乎相同。在估计精度和实时性能之间取得平衡，选择 N = 1024，并在对其他数据集训练的实验中应用这个数量的采样点。通过测试不同采样点表明，该方法对少量采样点具有鲁棒性，对大量采样点鲁棒性不明显，因为当 N = 512 时性能不会下降很多，而当 N=2048 时，性能并没有大幅的提升。

3.4 深度图像转换点云处理

PointNet 是直接对点云进行分割处理的方法，而通过 Realsense 深度相机采集的深度图像需要先进行点云转换后方能用于数据输入。

深度图转换为点云是 3D 点投影到 2.5D 曲面的逆过程。如图 3-11 为三种不同的坐标系。

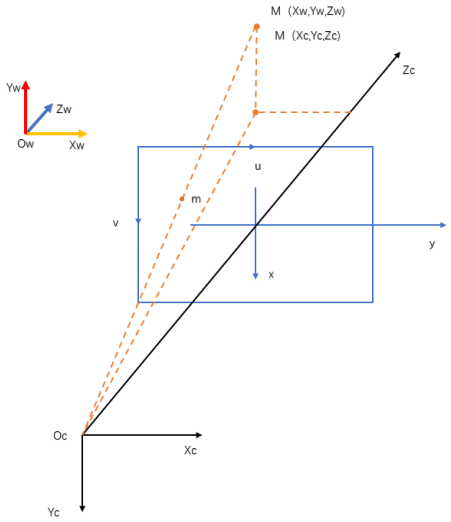


图 3-11 三种坐标系

世界坐标$M(x_w, y_w, z_w)$到图像点$m(u, v)$的映射见公式（3-4）：

$$Z_c \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{dx} & 0 & u_0 \\ 0 & \frac{f}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot [R \quad T] \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{3-4}$$

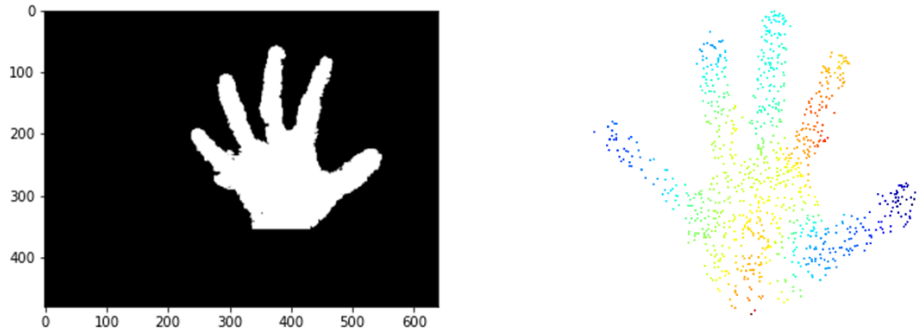
通过深度摄像头获取手势的深度值，将深度值以毫米为单位存储在 16 位无符号整数中(0~65535)，如果要获得以米为单位的 z 值，深度图的像素需要除以比例因子 1000。

$$scalefactor = depth_map_value \,/\, real_depth \tag{3-5}$$

比例因子根据不同相机的型号可能不同，比如 TUM 数据集的比例因子为 5000，即深度图中存储的深度值的单位为 200 毫米。

点云（point cloud）通常是三维空间的点集，每个点可以包含 x，y，z，颜色、分类值、强度值、时间等信息，是 3D 数据的表示形式之一 [19]。

由 Intel 发布的 open3D 库能够很好的配合 Realsense 深度相机实现深度图像转换点云。就像 OPENCV 是 2D 图像的处理库一般。转换前后的深度图灰度图像和点云图像如图 3-12 所示：



(a) 深度图像　　(b) 点云图像

图 3-12 点云转换可视化结果

在深度图像采集中，首先通过相机距离阈值判断对手部之后的背景进行去除，保留手势部分，然后通过 open3D 从 4 通道 RGBD 图像中读取深度信息，同时$profile.get_stream(rs.stream.color).as_video_stream_profile().get_intrinsics()$读取相机内参：图像分辨率为$640 \times 480$，光学中心$(ppx, ppy) = (328.281, 250.802)$，焦距$(f_x, f_y) = (605.827, 604.816)$。然后使用 open3D 的 $create_from_color_and_depth$ 函数将彩色图和深度图转换成 RGBD 图，彩色图是 8 位深度三通道，深度图是 16 位深度，最后通过 RGBD 图生成点云。

3.5 PointNet 手势关键点识别

通过 Realsense 深度相机采集深度图像，通过 open3D 生成点云并进行点云配准后得到的点云，用 PointNet 对手势点云进行处理，得到 21 个手部关键点位置和深度信息，用于仿生手臂控制。为了能通过 Pointnet 网络推理出 21 个手部关键点，需要先进行预处理生成点云方向向量，然后通过 OBB 对点云进行旋转后输入生成1×63的关节点数据，最后按照预处理逆流程，反向计算关节点的世界坐标。

3.5.1 OBB 转换

3D 手的位姿估计的一个难点就是手的全局方向变换很大。通过对手势点云归一化的目

的是为了将原始的手的坐标转到一个典型的坐标系统中，使手势方向和训练的模型方向尽量一致。这个归一化的目标是为了确保在对点云输入进行处理时面对手的全局旋转时更加具有鲁棒性。

如果输出对输入点云的旋转是不变的，例如语义标记任务中的输出，可以在 PointNet 中添加一个空间变换网络来预测变换矩阵[20]。然而，在点云输入的情况下，输出的 3D 手部姿势取决于输入手点云的方向，如果输入的手势点云方向的任意的，直接通过网络估计的手势误差比较大[21]。

本文通过一种简单而有效的方法来标准化 OBB 中的 3D 手点云，而不是应用任何额外的网络来估计手的全局方向或变换手势点云。OBB 是输入点云的紧密拟合包围盒。OBB 的方向是通过对输入点的 3D 坐标确定的[22]。首先将相机 C.S. 中的原始点转换为 OBB C.S.，然后将这些点移动到均值为零并缩放为单位大小，见公式（3.14）、（3.15）所示：

$$\mathrm{p}^{obb} = (R_{obb}^{cam})^{T} \cdot \mathrm{p}^{cam} \tag{3-6}$$

$$p^{nor} = \frac{p^{obb} - p^{obb}}{L_{obb}} \tag{3-7}$$

R_{obb}^{cam}是相机中 OBB 的旋转矩阵，p^{cam}为 p 在深度相机坐标系下的坐标，p^{obb}为 OBB 归一化后的坐标，$\bar{p}^{obb}$为形心坐标，L_{obb}为边界框的最大边长，p^{nor}为标准化 OBB 坐标系中最终的归一化坐标。

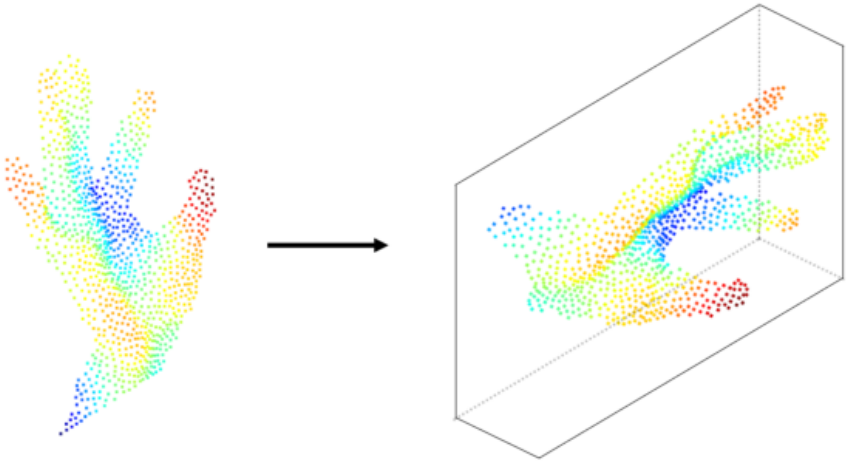


图 3-13 OBB 转换点云

图 3-13 展示了一些原始的手势点云和相应的归一化后的手势点云的对比。可以看出，由于人手的角度变化大，导致通过深度相机直接采集的深度手势图像的方向差异性较大，很难做到一致，通过 OBB 进行归一化旋转后可以使手势点云的方向保持一致。

左边显示的为深度相机中的原始点云，具有各种手部方向。右边显示的为 OBB 归一化后采样和旋转的点云，其中 OBB 手势的方向更加一致。

3.5.2 PointNet++三维手势估计网络结构

PointNet++三维手势估计网络如图 3-14 所示：

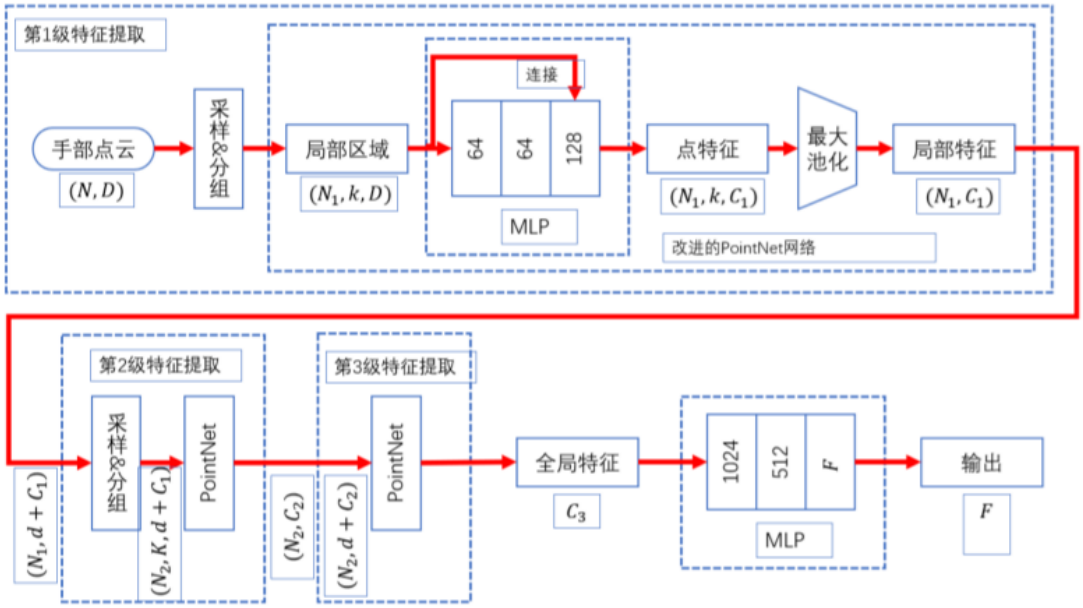


图 3-14 三维手势估计网络结构

PointNet++三维手势估计网络是基于 PointNet++网络，它利用层次网状网络进行 3D 手姿势估计。PointNet++三维手势估计网络的输入是一组归一化点 PO，见公式（3-8）所示：

$$PO = \{po_i\}_{i=1}^{N} = \{p_i^{nor}, n_i^{nor}\}_{i=1}^{N} \tag{3-8}$$

p_i^{nor}是归一化后的点的坐标，n_i^{nor}是对应点的表面法向量。N 是归一化手势点云的点的数量，输入层次结构网络(Hierarchical PointNet)。

如图 3-14 所示，网络有三个层级，前两层，$N_1 = 512$，$N_2 = 128$，每一层分别提取了$C_1 = 128$，$C_2 = 256$个特征。

由于人手的自由度通常比手的关节的自由度$(3 \times M)$少。PointNet++输出的是一个 F 维数据$(F < 3 \times M)$。通过对自由度进行约束，降低手的位姿在不合理范围内的可能性。训练过程中，训练样本包括归一化的点云数据和 3 维真实关节坐标值$\{po_T^{nor}, gr_T^{nor}\}_{t=1}^{nor}$，最小化的目标函数见公式（3-9）所示：

$$\theta^* = \arg\min_{\theta} \sum_{t=1}^{T} \alpha_t - \gamma(po_T^{nor}, \theta)^2 + \lambda\left|\left|\theta\right|\right|^2 \tag{3-9}$$

通过在训练的数据集中的表面中心 3D 位置上 PCA，可以获得α_t见公式（3-10）所示：

$$\alpha_t = E^T \cdot (gr_T^{nor} - u) \tag{3-10}$$

其中 E 是主要成分，u是经验平均值。最后根据网络输出，按照处理逆流程，反向计算关节点的 3D 坐标见公式（3-11）所示。

$$\hat{g}^{nor} = E \cdot \gamma \cdot (po_T^{nor}, \theta^*) + u \tag{3-11}$$

第四章 机械手臂运动控制模块设计

本系统采用 RealSense D435 相机通过第三章所述方法获得人手部关节的坐标，通过一定的算法转换成机械手臂各舵机的旋转角度，从而操控机械手臂各自由度的运动。机械手臂硬件部分由六自由度机械手和六自由度机械臂组装而成，主控芯片采用高性能、低功耗的 STM32f103，主机与控制平台通过 WIFI 进行通信，并向机械手臂的主控芯片发送命令，实现对机械手臂的运动控制。

4.1 机械手模型与运动算法设计

4.1.1 机械手结构原理

图 4-1 六自由度仿生机械手实物

在本项目中采用深圳幻尔科技提供的六自由度仿生机械手，如图 4-2 所示。仿生机械手掌的五个手指由五个防堵转的 LFD-01 舵机驱动，腕部云台由 180 度的 LD-1501MG 数字舵机驱动。

LFD-01 舵机的如图 4-2（左）所示，舵机的旋转角度可以到达 0°~180°，并且舵机内置有防堵转算法，当发生堵转时，舵机可以自动卸力，保护舵机不会被烧坏，适用于各类仿生机器人关节的控制。

图 4-2 仿生机械手数字舵机

LD-1501MG 单轴数字舵机如图 4-2（右）所示，舵机的旋转角度可以到达 0°~180°，齿轮采用全金属设计，可以提供最大 15kg 的扭力，是一款专门用于机械臂关节控制的舵机。

每个舵机有三条控制线分别为 GND、供电和 PWM 信号线，如图 4-3 所示。

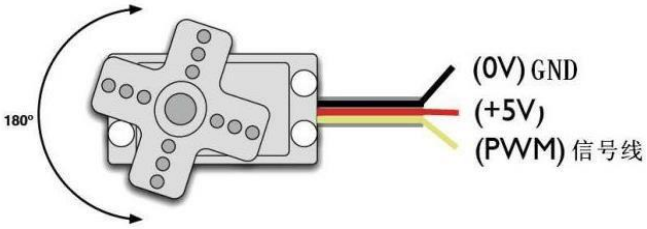


图 4-3 舵机控制原理图

其中电源线和地线用于供电，PWM 信号线的输入为脉冲信号。在一个 PWM 周期内，通过调整高电平占用的信号比例来调节舵机转动的角度位置，如表 4-1 所示。

表 4-1 输出脉冲高电平占比与舵机旋转角度关系

输出高电平占比（周期为 20ms）	舵机旋转角度位置
0.5ms	≈ 0°
1.0ms	≈ 45°
1.5ms	≈ 90°
2ms	≈ 135°
2.5ms	≈ 180°

主控芯片对仿生手的控制也就是给机械手的各个舵机发送不同占空比的 PWM 波的指令，舵机根据脉冲信号旋转到对应的角度。

4.1.2 机械手掌运动算法的设计

通过第 3 章对于手势的识别，可以得到 21 个手部关键点的三维坐标，如图 4-4 所示。21 个点分别对应一个手腕关键节点，和 5 个手指各 4 个关键节点，并依次从拇指到小指进行编号，按顺序将三维坐标进行保存。通过 4 个手指的上的关键点坐标，引出关键点射线，可分别计算出远指关节和近指关节两处的角度，两部分角度之和，即为对应的手指的控制舵机旋转角度。

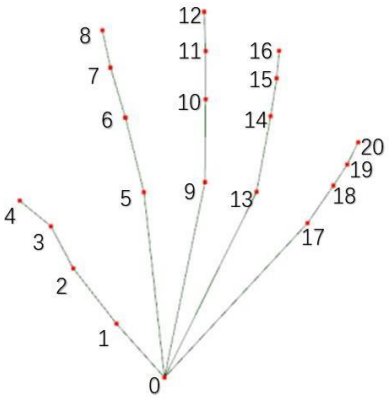


图 4-4 手势识别得到的 21 个关键节点

如图 4-5 所示，以食指部分夹角为例计算关节夹角。

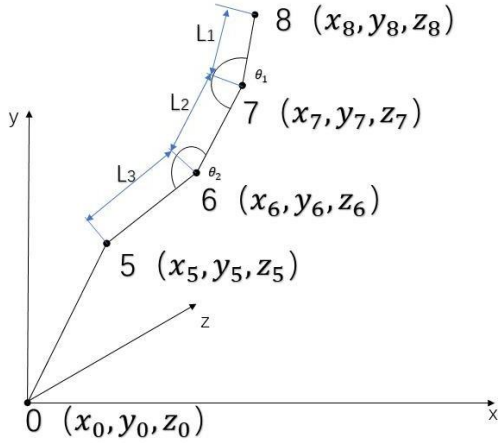


图 4-5 食指关节夹角计算

根据图 4-5 中 21 个关键节点中代表食指的点 5、6、7、8 的三维坐标值计算向量 $\overrightarrow{L_1}$ 、$\overrightarrow{L_2}$ 、$\overrightarrow{L_3}$，如下所示：

$$\overrightarrow{L_1} = \left(x_8 - x_7, y_8 - y_7, z_8 - x_7\right) \tag{4-1}$$

$$\overrightarrow{L_2} = \left(x_6 - x_7, y_6 - y_7, z_6 - x_7\right) \tag{4-2}$$

$$\overrightarrow{L_3} = \left(x_5 - x_6, y_5 - y_6, z_5 - x_6\right) \tag{4-3}$$

进而得到各向量在空间中的有向夹角：

$$cos\theta_1 = \frac{\overrightarrow{L_1} \cdot \overrightarrow{L_2}}{\left|\overrightarrow{L_1}\right| \cdot \left|\overrightarrow{L_2}\right|} (90° \le \theta_1 \le 180°) \tag{4-4}$$

$$cos\theta_2 = -\frac{\overrightarrow{L_2} \cdot \overrightarrow{L_3}}{\left|\overrightarrow{L_2}\right| \cdot \left|\overrightarrow{L_3}\right|} (90° \le \theta_2 \le 180°) \tag{4-5}$$

求得两个关节夹角之和：

$$\theta = \theta_1 + \theta_2 \, (180° \le \theta \le 360°) \tag{4-6}$$

最后通过角度与舵机映射关系转换为舵机 PWM 波形高电平占比，如图 4-6 所示。

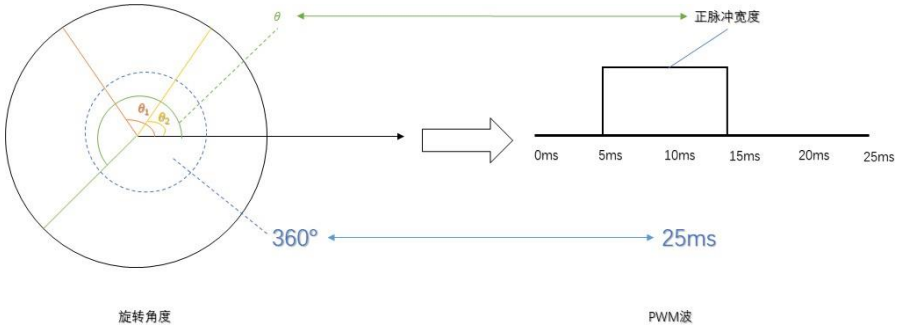


图 4-6 旋转角度与 PWM 波映射

为了能够使得机械手能够模仿人手在竖直平面和水平平面内旋转，需要对机械手的底座舵机进行控制。这里采用计算人手掌所在的平面法向量与相机坐标系的坐标平面的角度差，来实现机械手在空间内的旋转。

当人手指完全伸展时，通过手势估计得到的如图 4-4 的 21 个点，所拟合的平面即为人手掌所在的平面，显然，该手掌平面的法向量应仅与人手腕部的旋转有关，而不应随手指弯曲程度变化而变化。于是考虑到当人手指弯曲到最大程度后，人手指的指尖、远、近端指间关节，都会明显地落在人手掌平面外，其坐标变化很大。相较而言，人手指的掌指关节和腕部关节仍然落在手掌平面内，坐标基本没有变化，因此选取人手指的掌指关节坐标和腕部关节坐标拟合出人手掌所在的平面。考虑到大拇指在完全弯曲后掌指关节坐标仍有一定明显的变化，我们选取图 4-4 中的点 0、5、9、13、17 作为离散点，通过最小二乘法，拟合出手掌所在的平面。

设所取的五个点坐标分别为$P_i(x_i, y_i, z_i)(i = 1,2,\dots,5)$，所要拟合的平面解析方程为：

$$A \cdot x + B \cdot y + C \cdot z + D = 0 \quad (C \neq 0) \tag{4-7}$$

即：

$$z = -\left(\frac{A}{C} \cdot x + \frac{B}{C} \cdot y + \frac{D}{C}\right) \tag{4-8}$$

记$a_0 = -\frac{A}{C}$，$a_1 = -\frac{B}{C}$，$a_2 = -\frac{D}{C}$，则式可化为：

$$z = a_0 \cdot x + a_1 \cdot y + a_2 \tag{4-9}$$

要用$P_i(i = 1,2,\dots,5)$这五个点拟合平面方程，即使得：

$$S = \sum_{i=1}^{5} (a_0 \cdot x_i + a_1 \cdot y_i + a_2 - z_i)^2 \tag{4-10}$$

最小，故应满足：

$$\frac{\partial S}{\partial a_k} = 0，\ \mathrm{k} = (0,1,2) \tag{4-11}$$

即：

$$\sum_{i=1}^{5} 2 \cdot (a_0 \cdot x_i + a_1 \cdot y_i + a_2 - z_i) \cdot x_i = 0 \tag{4-12}$$

$$\sum_{i=1}^{5} 2 \cdot (a_0 \cdot x_i + a_1 \cdot y_i + a_2 - z_i) \cdot y_i = 0 \tag{4-13}$$

$$\sum_{i=1}^{5} 2 \cdot (a_0 \cdot x_i + a_1 \cdot y_i + a_2 - z_i) = 0 \tag{4-14}$$

改为矩阵形式：

$$\begin{vmatrix} \sum {x_i}^2 & \sum x_i y_i & \sum x_i \\ \sum x_i y_i & \sum {y_i}^2 & \sum y_i \\ \sum x_i & \sum y_i & 5 \end{vmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum x_i z_i \\ \sum y_i z_i \\ \sum z_i \end{bmatrix} \tag{4-15}$$

解方程组，即可求得参数a_0、a_1、a_2，进而求得平面方程$A \cdot x + B \cdot y + C \cdot z + D = 0$，该平面的单位法向量即为：

$$\vec{n} = \frac{(A, B, C)}{\sqrt[2]{A^2 + B^2 + C^2}} \tag{4-16}$$

设该拟合平面与水平平面的夹角为α，等于：

$$\alpha = arcos\left(\frac{\vec{n}\cdot\overrightarrow{n_1}}{|\vec{n}|\cdot|\overrightarrow{n_1}|}\right) \quad (4-17)$$

其中$\overrightarrow{n_1} = (0,1,0)$，为相机坐标系平面$xOz$的法向量，则其在竖直平面内的旋转角度为$90° - \alpha$。同样地，设该拟合平面与竖直平面$xOy$的夹角为$\beta$，等于：

$$\beta = arcos\left(\frac{\vec{n}\cdot\overrightarrow{n_2}}{|\vec{n}|\cdot|\overrightarrow{n_2}|}\right) \quad (4-18)$$

其中$\overrightarrow{n_2} = (1,0,0)$，为相机坐标系平面$yOz$的法向量，则其在水平平面内的旋转角度为$\beta$。

在实际工程中，考虑到程序运行时占用的计算资源、用户操控的便捷性以及可靠性，在以上的算法上进行一定的约束与优化，从而使得对机械手的控制更为舒适、流畅。

在上述算法中，为了使机械手能够模仿人手腕部的旋转，我们通过取手上五个点拟合成手掌平面，而后计算平面法向量与坐标平面的角度差，该角度差即为机械手掌腕部舵机应旋转的角度。但这里取五个离散点，通过最小二乘法拟合平面求解略显复杂，考虑到三个离散点即可形成一个平面，以及人手指最大弯曲后掌指关节预测坐标的变化，可以仅取图 4-5 中 0、5、13 这三个关键节点，设为点P,A,B，取这三个点形成的两个不同向量$\overrightarrow{PA},\overrightarrow{PB}$，则人手掌平面的法向量为$\vec{n} = \overrightarrow{PA}\times\overrightarrow{PB}$，而后由公式（4-17）和（4-18）即能计算出旋转角度，极大地节省了计算量。

4.2 机械臂模型与运动算法设计

4.2.1 机械臂模型建立

机械臂同样采用幻尔科技提供的六自由度仿生机械臂，通过用机械臂的腕部舵机取代机械手的腕部舵机的方式， 能够将该机械手和机械臂连接成一个整体，如图 4-7 所示。和机械手相似，该机械臂使用全铝合金结构作为金属结构，搭载不同的数字舵机驱动，使用 STM32 作为主控制板。其末端关节舵机采用防堵转设计，大大提高了机械臂整体的动力。其舵机型号分别为：LD-1501MG 舵机、LDX-218 舵机、LFD-06 舵机、QDS-15 舵机，前两种舵机为大扭力高精度舵机，用于底座和底部关节；后两种舵机则带有防堵转设计并带有过流保护，用于末端关节和手掌。

图 4-7 机械手臂组合实物

从机构运动学分析，机械臂主要包含关节和连杆两部分，对于机械臂的运动状态研究，首先需要确定机械臂的连杆个数和关节个数，然后确定各关节连杆之间的运动关系，最后通过基座的位置确定末端执行器的位姿变化。其模型如图 4-8 所示：

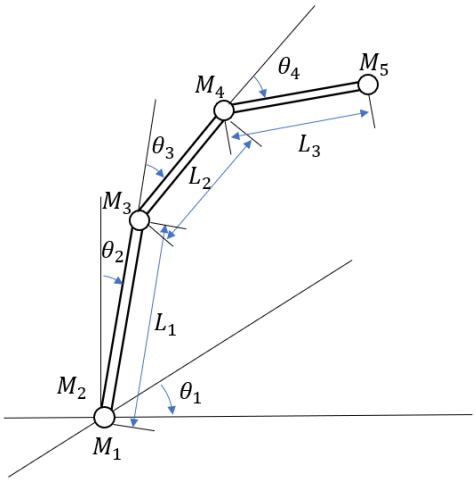


图 4-8 机械臂模型

表 4-2 机械臂模型参数取值

L_1	L_2	L_3	θ_1	θ_2	θ_3	θ_4
13cm	9cm	12cm	$-90°\sim90°$	$-90°\sim90°$	$-90°\sim90°$	$-90°\sim90°$

4.2.2 机械臂运动算法设计

机器人逆运动学是由末端位姿的变化来计算各个机械臂连杆之间的角度信息变化，其分析方法有解析法、迭代法和几何法，由于本项目所使用的机械臂模型结构相对简单，因此使用几何法来求解。

设相机坐标系内人手腕部的初始点坐标为E_0，当前点坐标为E,则其位移为向量$\overrightarrow{E_0E}$。在机械臂坐标系内，原点O为底座舵机M_1的位置。设机械臂末端为舵机M_5，其初始点坐标为P_0，当前点坐标为P,则其位移$\overrightarrow{P_0P}=\lambda\overrightarrow{E_0E}$，其中$\lambda$表示伸缩系数，其含义为当人手腕部移动$l$ cm 时，机械臂末端移动$\lambda\cdot l$ cm。考虑点P位于x轴正半轴的情况。

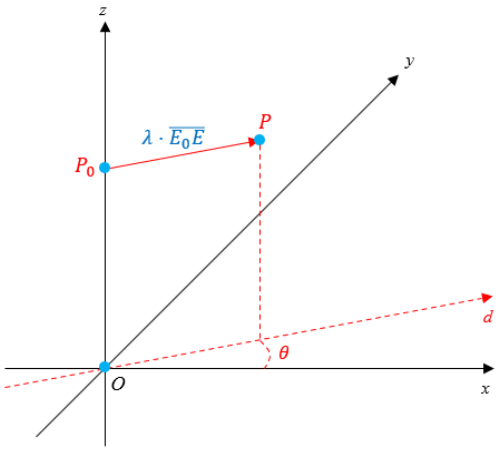


图 4-9 机械臂末端在机械臂坐标系内移动示意图

设点 P 坐标为(x_0、y_0、z_0),则|OP| $=\sqrt[2]{{x_0}^2+{y_0}^2+{z_0}^2}$。如图 4-10 所示，$\theta=arctan(\frac{y_0}{x_0})$即为机械臂 6 号舵机(关节$M_1$)旋转角度。

以射线d作为新的 x 轴，z 轴不变建立新的平面坐标系。在该坐标系中，将机械臂关节M_2、M_3、M_4视作质点，标注为点O、A、B，机械臂末端当前位置为点P,线段OP长度不变，仍为$\sqrt[2]{{x_0}^2+{y_0}^2+{z_0}^2}$。由于点P在平面$xy$的投影位于射线$d$上，故在新的坐标系中点P

与x轴的夹角等于点P与平面xy的夹角，其值$\sigma = arctan(\frac{z_0}{{x_0}^2+{y_0}^2})$。

OA、AB、BP的长度分别为机械臂各杆长，即$\ell_1 = L_1$、$\ell_2 = L_2$、$\ell_3 = L_3$，所求各舵机旋转角度为α、β、γ。以点O为圆心，ℓ_1为半径做圆，该圆为机械臂连杆L_1所能活动的范围，则点A位于该圆上；以点P为圆心，ℓ_3为半径做圆，该圆为机械臂连杆L_3所能活动的范围，则点B位于该圆上。线段AB即为机械臂的连杆L_2，故有$|AB| = \ell_2$。能够使$|AB| = \ell_2$的圆上的点A和B有无数个，考虑到项目所使用的机械臂连杆活动范围为180°，同时为了连杆间角度较为均匀，不出现过大的角度使得机械臂所承受的压力过大，故连接OP，令AB平行于OP。延长OA、PB交于点C。

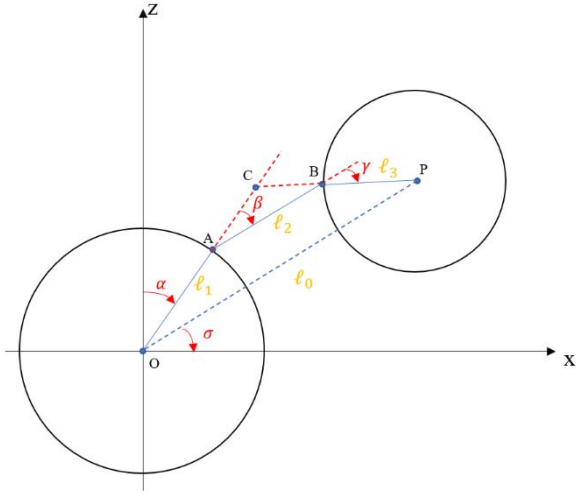


图 4-10 在$x'(d) - z$坐标系内求解示意图

根据***梅涅劳斯定理***，有：

$$\frac{|CA|}{|CO|} = \frac{|CB|}{|CP|} = \frac{|AB|}{|OP|} \tag{4-19}$$

又根据$|CO| = |CA| + |AO| = |CA| + \ell_1$；　$|CP| = |CB| + |BP| = |CB| + \ell_3$，故得：机械手臂

$$|CB| = \frac{\ell_3 \cdot \ell_2}{\ell_0 - \ell_2} \tag{4-21}$$

则在ΔABC中，运用余弦定理：

$$\cos\beta = \frac{|CA|^2 + |AB|^2 + |CB|^2}{2|CA| \cdot |AB|} \tag{4-22}$$

$$\cos\gamma = \frac{|CB|^2 + |AB|^2 + |CA|^2}{2|CB| \cdot |AB|} \tag{4-23}$$

通过反余弦函数求得β、γ，又因为AB $\parallel$ OP，故$\angle AOP = \angle CAB = \beta$，又

$\sigma = arctan(\frac{z_0}{{x_0}^2+{y_0}^2})$为射线$\overrightarrow{OP}$与$x$轴夹角，则有：

$$\alpha = 90° - \beta - \sigma \tag{4-24}$$

当点P落在x轴负半轴的情况按照如上方法同理可得：

$$\alpha = \sigma - 90° - \beta \tag{4-25}$$

同样地，考虑实际工程中人手牵引机械臂的可操控性，也对上述算法的加以一定的约束和优化。首先，当人手对机械手臂进行牵引时，腕部很难维持在相机坐标系中央的竖直平面，并且当人手腕部移动距离较小时，该晃动会对θ产生很大的影响。因此需要对θ和机械臂 6 号舵机实际旋转角度重构映射关系。

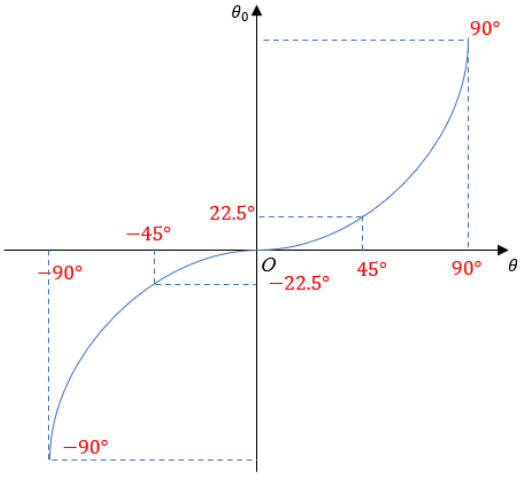


图 4-11 θ_0与θ映射关系

为了使得θ角度较小时，机械臂 6 号舵机实际旋转角度较小（用以减小手腕部晃动对操控的影响），这里采用分段二次函数来重新定义这种映射关系。设机械臂 6 号舵机实际旋转角度为$\theta_0(-90° < \theta_0 < 90°)$，则有$\theta_0 = \frac{|\theta|^3}{90 \cdot \theta}$，$\theta_0$与$\theta$同符号。

其次，考虑到人手不可避免地在一定范围内进行晃动时，当其在原点处晃动时，坐标的变化对于机械臂底座舵机旋转角度影响很大，即θ_0变化很大，而此时我们并不期待θ_0有变化。因此当且仅当人手腕部实际位移$\overline{E_0E} > 1$时，才依据公式计算出θ_0，否则令$\theta_0 = 0$。

4.3 仿生手臂控制系统硬件设计

仿生手臂的控制芯片采用基于 Cortex-M3 内核的 STM32 单片机，它具有功耗低，处理性能优越，中断响应时间短，代码密度高，价格低廉等优势，非常适合作为机械手臂的控制平台。在本系统中，机械手和机械臂的主控芯片选择型号为 STM32F103VCT6 的芯片，采用 ARM® 32-bit Cortex®-M3 CPU 内核，主频最大可达 72MHz，含有 256K 的内置 FLASH 和 64K 的内置 SRAM。在保证机械手臂的实时响应要求的前提下，极大地降低了控制核心的成本，并提供了极高的数据处理速率。

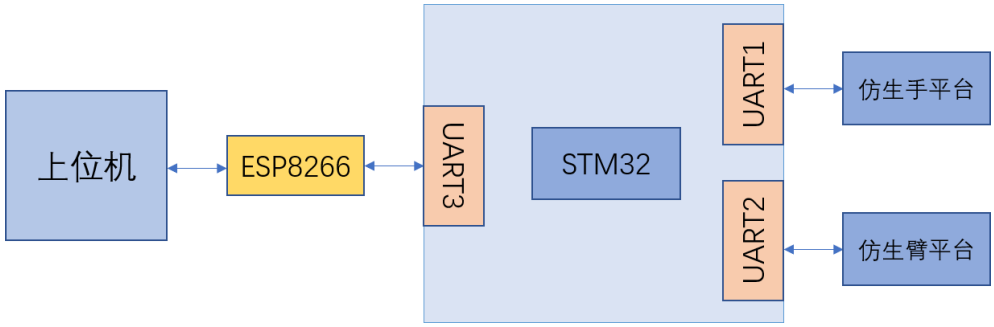


图 4-12 系统通信框图

系统采用模块化控制，以一片 STM32 为核心，通过 ESP8266 模块建立 TCP 连接，实现与主机的远程通信功能。控制平台分别通过 2 个 UART 模块与机械手与机械臂的主控芯片进行通信，模块之间采用了特定的通信协议以保证数据传输的可靠性。

在 **4.1** 和 **4.2** 中，我们得到了机械手臂各个舵机旋转的角度，上位机通过特定的通信协议将该角度数据进行封装，再通过网络发送到控制平台，控制平台解码该数据包并将命令发送给相关的主控芯片。机械手和机械臂的主控芯片接收角度数据，通过角度与 PWM 脉冲的映射，将其转化为输出脉冲中的高电平占比，实现舵机特定角度的旋转，完成对仿生

手臂控制。

在控制板与上位机的通信中，规定了上位机以 5 帧/秒的左右的速率向终端机发送数据帧，每一帧由帧头、帧尾以及 A1~A5 的数据部分构成，每一个部分的具体作用如下：

表 4-3 控制平台与上位机 WIFI 通信协议

帧头	A1	A2	A3	A4	A5	帧尾
帧头：固定数据为”dd”，标志每一帧的开始。收到帧头后接受 5 位数据作为数据位						
A1：数据为“0”时与机械臂控制端通信；数据为“1”时与机械手控制端通信。						
A2：选择机械臂/机械手的 1~6 号舵机						
A3、A4、A5：控制旋转角度 0~180°						
帧尾：固定数据为“ff”，标志每一帧的结束。						

控制平台与终端主控芯片之间采用了不同结构的帧进行通信，具体如下：

表 4-4 终端主控芯片与控制平台串口通信协议

帧头	B1	B2	B3	B4	B5	B6	B7	B8
帧头：固定数据为”dd”，标志每一帧的开始。收到帧头后接受帧头后的 5 位数据作为数据位								
B1：数据长度，值应为控制舵机数*3+5								
B2：填充舵机移动指令；								
B3：要控制的舵机个数；								
B4：填充运动时间的低八位								
B5：填充运动时间的高八位								
B6：填充舵机 ID								
B7：填充目标位置的低八位								
B8：填充目标位置的高八位								

其中 B6~B7 仅表示第一个舵机的控制信息，如控制多个舵机则其余控制信息按照相同的格式在帧尾部续写直至所有信息发送完成，因此帧的长度不定。

上位机向控制平台发送的信息可以抽象为目标舵机号和旋转角度，控制平台将旋转角度的信息转译为目标位置信息并将信息发送到对应的终端，终端再将位置信息转译成每个舵机的旋转方向与运动时间，控制舵机旋转从而使机械臂/机械手结构到达预定位置。

第五章 移动平台的 SLAM 建模与远程控制模块设计

在系统整体设计中，我们设置了一个用于运载机械手臂的移动平台，平台总体设计如下图所示。

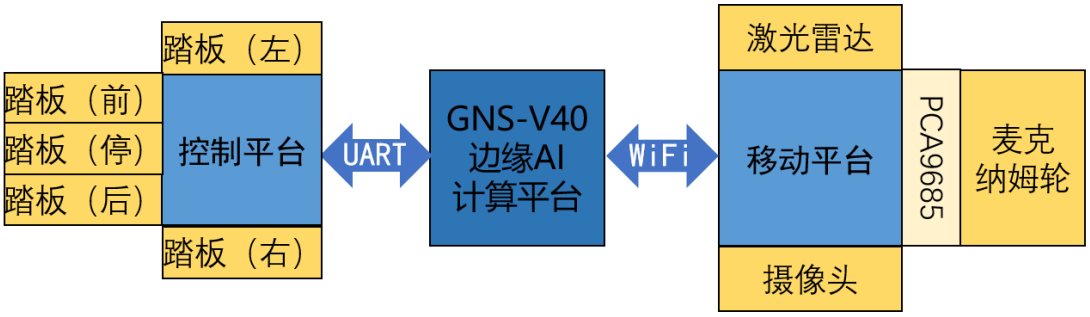


图 5-1 基于 STM32F103 的移动平台远程控制设计方案图

该平台底部配有四只麦克纳姆轮，承重 40kg，可以满足运载机械手的需求。平台配有瑞芯微工控主机，处理器为 RK3399，操作系统为 Ubuntu 16.04，基于 ROS 实现激光雷达 SLAM 建模，配合摄像头可用来观察移动平台周围的环境。在赛会提供的计算平台上，可通过踏板远程控制平台移动，并通过 SSH 的方式获取 SLAM 建模以及摄像头的画面，实现远程控制的功能。

5.1 基于 STM32F103 的移动平台远程控制设计与实现

平台移动依靠 PCA9685 驱动电机实现。PCA9685 芯片可以输出多达 16 路的 PWM 波，其中每两路 PWM 波就可以控制 YX4055-AM 驱动电机正转、反转或者停止，因此使用一只 PCA9685 就可以控制四只麦克纳姆轮，这极大地节省了微控制器资源以及电路板的空间。在移动平台中，我们设置不同的参数，向 PCA9685 芯片中对应的地址设置占空比数值，从而向对应的电机输出相应频率的 PWM 波，不同频率的 PWM 波则对应着不同的转速，若需要反转，则将 PWM 波反相即可。

ESP8266 是一个常见的 Wi-Fi 模组，内置 TCP/IP 协议，借助于 UART 通信协议任意微控制器可以通过 Wi-Fi 与 ESP8266 传输数据。在本项目中，ESP8266 被配置为 AP 模式，在该模式下，模组可以接收上位机传输过来的数据并将数据传输给客户端。我们使用 Python 搭建服务器接收微控制器传输的控制信息，并将控制信息打包成 TCP 报文发送至 ESP8266 模组中，再由该模组向移动平台传输控制信息，平台在收到 TCP 报文后，解析报文中的控制信息经由 PCA9685 控制麦克纳姆轮带动小车移动。

此外，在边缘计算平台上，操作人员可以远程实时观察机械手臂的作业情况，效果如图 5-2 所示。

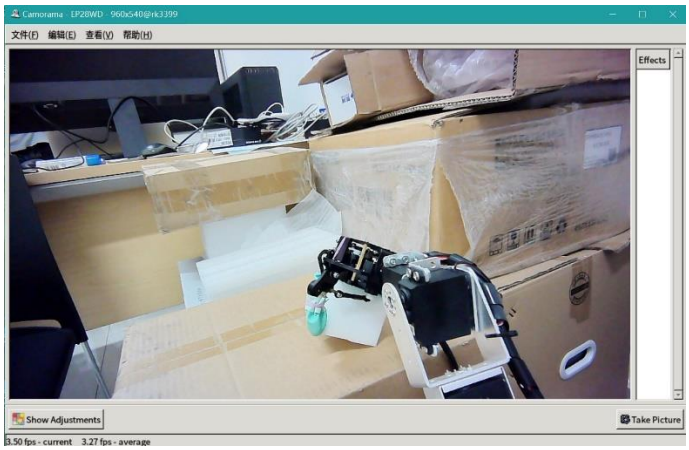


图 5-2 作业平台实时画面

5.2 基于 ROS 的激光雷达 SLAM 建模

SLAM，全称 Simultaneous Localization and Mapping，即同时定位与绘图，该技术可以被阐释为：移动机器人在未知环境下依靠传感器设备获取环境信息，并利用这些环境增量式地构建环境地图和自主定位。团队所选取的激光雷达采样率可达 8000 Hz，具有 16m 的测距半径，因项目主要应用于室内，其性能已经足够使用。

在诸多实现 SLAM 的算法中，项目团队选取了 Hector SLAM 这一算法建模。该算法基于扫描匹配进行建模，即使用当前帧与已有地图数据构建误差函数，然后使用高斯牛顿法得到最优解和偏差量，实现激光点到栅格地图的转换，建模成功的标志为某一时刻所有的激光点都能变换到格栅地图中。

在 ROS 中，Hector SLAM 相关算法已经被集成到其软件源中，在 Linux 操作系统中通过命令即可安装。场地的 SLAM 建模效果如图 5-3 所示：

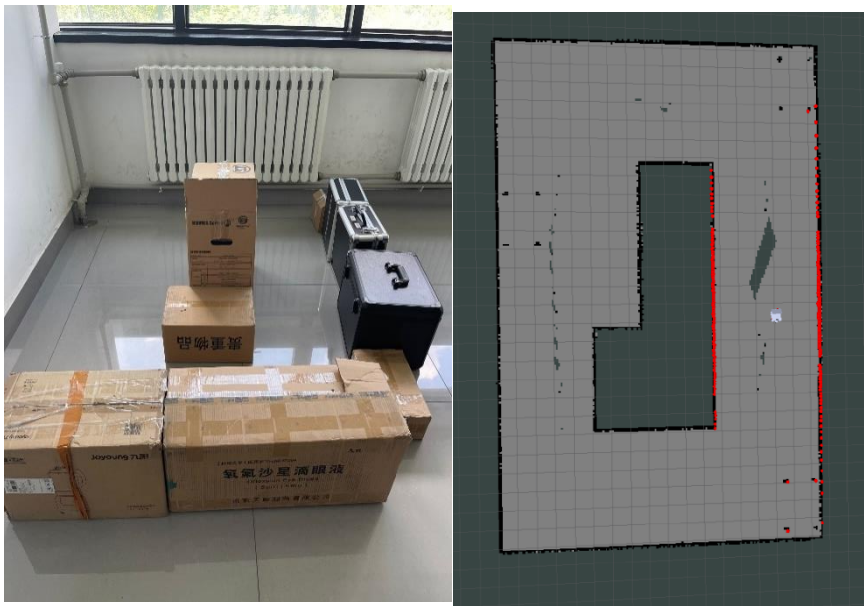


图 5-3 SLAM 建模效果图

图中建模场地为团队实验场地，可以清楚地看到激光雷达反馈的红色扫描点以及灰色匹配地图，对比场地布置障碍物，SLAM 建模可以清楚的分辨障碍物。其建模地图较为稀疏，不适用于高精度的地图建模，但是在本项目中，SLAM 主要用于观测移动平台周围环境，这样的建模效果已满足需求。

第六章 系统测试

6.1 RGB-D 手势识别算法测试

手势识别方法有 2D 图像，基于点云转换为 3D 体素后处理的方法和直接基于三维点云的方法。

数据集中数据量的大小和数据的准确性，对于训练出的网络检测手势的效果有着重要影响，本文通过 MSRA15 手势数据集从测试集检测手部关节估计的平均距离误差与其他不直接基于点云处理的 3D 深度卷积方法进行比较。不同方法在 MRSA15 下的平均距离误差如表 6-1 所示。

从表 6-1 中可以看出，与很多基于 2D 深度图像或者体素的间接方法在平均误差上都没有基于直接对点云进行处理的 PointNet 方法低。并且通过使用 OBB 对点云输入前先进行归一化，将点云全局旋转到一个典型坐标系下在输入网络，以及在检测手指指尖时通过设计一个指尖优化网络对指尖附近邻近点采样估计的方式，可以更好的降低误差。

表 6-1 MRSA15 手势数据集关节平均误差

深度图像手势估计方法	手势关节平均误差/mm
Multi-View-CNN	13.10
improved-3DCNN	10.60
3D CNN	9.60
DeepPrior++	12.51
PointttoPoint	9.10
Pose-REN	8.65
PointNet	8.70

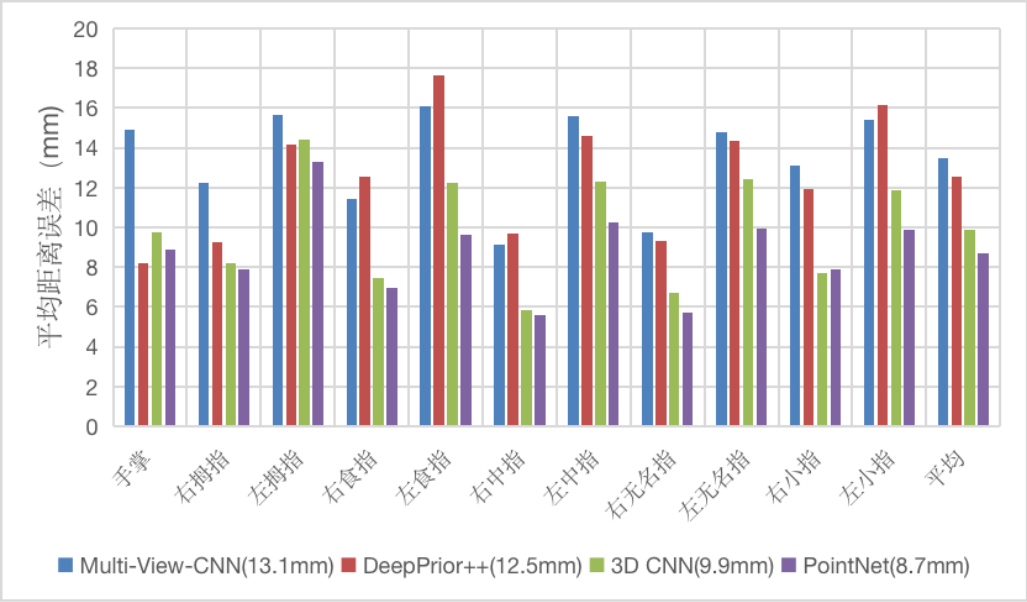


图 6-1 PointNet 网络关节点平均距离误差

在 MSRA15 的测试集上得到每个关节平均距离误差，并与其他方法进行比较，如图 6-1 所示。从图 6-1 中可以看出，与其他方法对比，PointNet 点云网络在手势各关节上的识别的平均距离误差都更低。

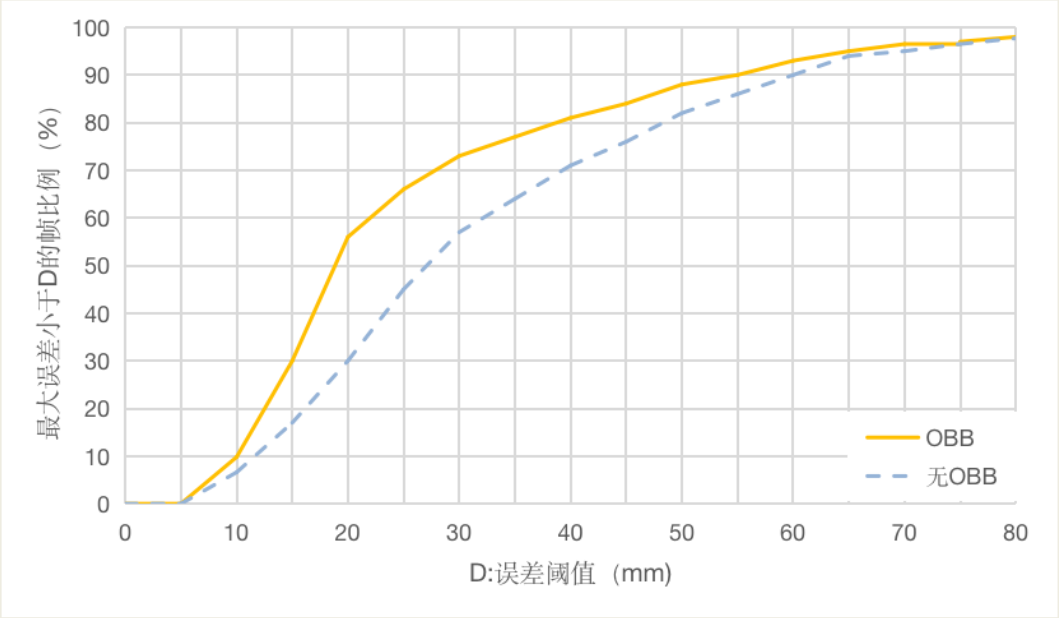


图 6-2 OBB 归一化方法对错误帧比例

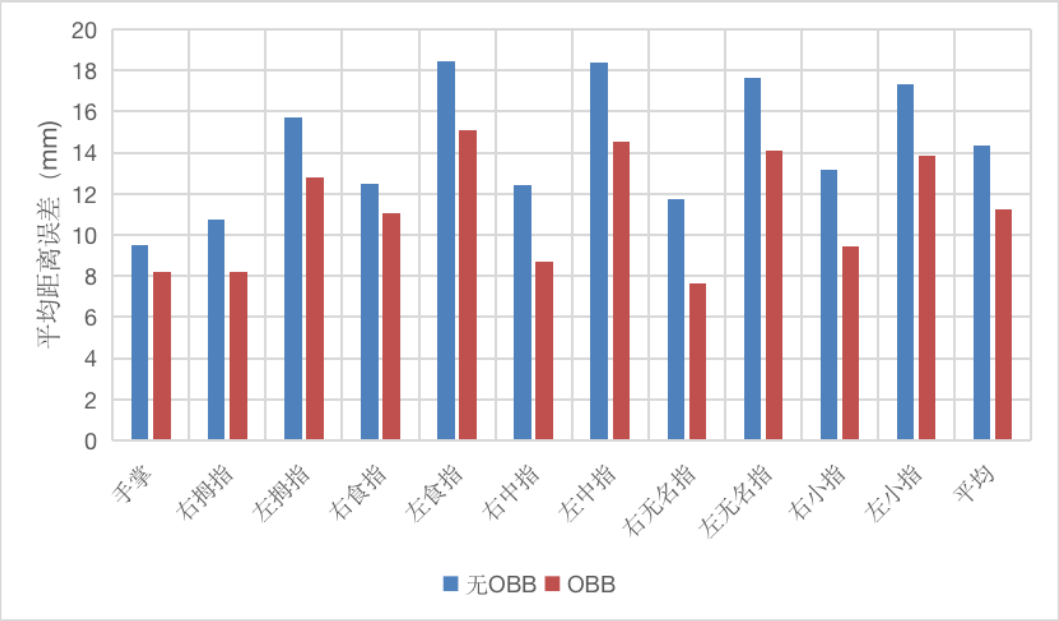


图 6-3 OBB 每个关节平均误差距离的影响

测试 OBB 归一化方法对手势模型训练的影响，对于不使用 OBB 的归一化方法，将点云中心移至零并将点缩放为单位球体而不旋转点云。如图 6-2 和图 6-3 所示，在相同的 PointNet 训练方法下，在使用 OBB 进行手势点云旋转后识别的手势准确性比直接进行识别的手势准确性更高，这表明对于 PointNet 使用点云识别手势时，使用 OBB 的点云归一化对点云手势进行旋转处理后识别手势的效果非常有效，因为它可以对具有更一致方向的点云进行归一化，并使网络更容易学习手部关节。此外，通过图 6-3 的每个关节的平均误差距离的影响图可以看出，当采用相同的 OBB 归一化方法时，分层 PointNet 在估计精度上的表现优于基本 PointNet。

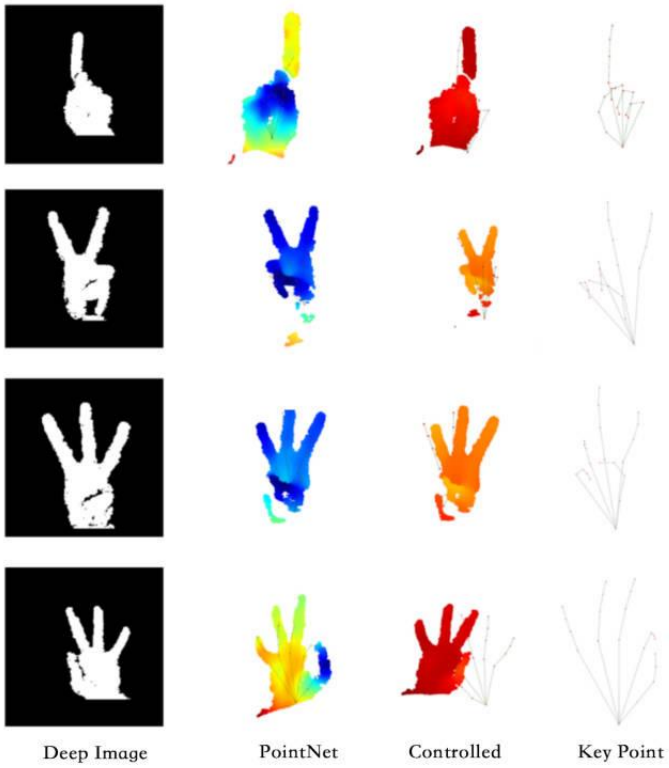


图 6-4 PointNet 关节点识别

同时针对不同背景下的不同手势进行多次测试，如图 6-4 所示。与基于 2D 图像的 Medipipe 方法比较，测试点云在复杂环境下通过包含更多的深度数据进行的手势估计效果与从 2D 图像中估计出的手势关节点的对比。从图中可以看出，在光线和环境相对比较复杂的实际环境中，两种方法在手指相互遮挡部分不大时，都能比较准确的从中手势中估计出关节点。但是，基于 2D 图像的 Mediapipe 方法在手指存在相互遮挡部分比较大时，关键点估计误差就比较大。而由于点云可以的包含更丰富的信息，通过点云进行手势估计的 PointNet 方法在手指关节遮挡比较严重时，也能比较准确的估计出手部关键点。

6.2 仿生机械手控制测试

由手势识别得到的手部关键节点信息，控制仿生机械手做出相应的手势。如图：

图 6-5 实时手势控制机械手效果图

图 6-6 实时手势控制机械手效果图

由图 6-5 和图 6-6 可以看出，系统可以很好地根据识别出的手部关键节点信息，控制仿生机械手做出相同姿势，兼具实时性和准确性。

6.3 系统测试

测试员充当用户，使用踏板操控远程移动平台移动到作业区，并用手势操控仿生机械手臂完成指定动作，结果如下：

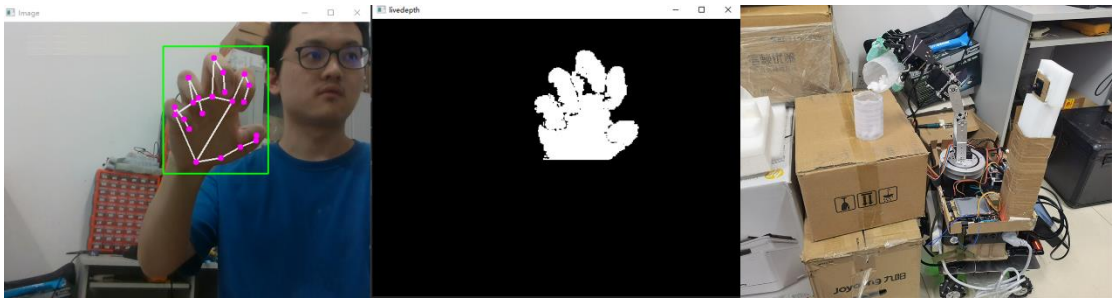


图 6-7 远程控制作业平台倒水演示

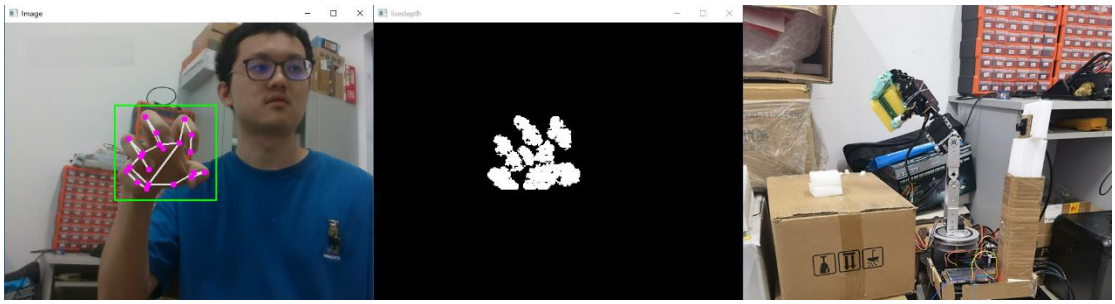


图 6-8 远程控制作业平台取物演示

利用该系统，食用者可以远程控制平台进行作业。如图 6-7 和图 6-8 所示，可控制该平台完成一些日常动作，如倒水、取物等。

第七章 关键技术及创新点

7.1 系统功能实现

表 7-1 系统功能实现情况

功能	实现情况
实时动态三维手势识别	实现
视觉牵引仿生机械手臂	实现
远程控制完成日常动作	实现
实时图像获取	实现
SLAM 定位	实现
作业平台移动控制	实现

7.2 系统特色与创新点

（1）实现实时的、动态的手势识别，其预测的手部关键节点精度较高，误差不超过8.4mm。

（2）通过手势远程控制仿生机械手臂运动，能够较为完美地复现人手的姿态，并牵引机械手臂完成日常动作，如取物、倒水等，兼具低延迟、稳定性。

（3）通过踏板远程控制作业平台移动，由部署在平台的摄像头实现实时场景跟踪，并通过 SLAM 实现作业平台定位。

（4）系统高速度、低延迟。手势识别模块可以实现每秒 10 帧的实时处理速度，远程控制仿生机械手臂可以实现 200ms 以内的延迟。

（5）系统采用模块化设计，可以根据不同的应用场景来改进各模块，具有较高的适用性。

7.3 系统关键技术

（1）基于 PointNet++网络的高精度、实时、动态手势识别。

（2）实现通过手势控制的仿生机械手臂运动算法。

（3）实现远程控制机械手臂运动、作业平台移动和 SLAM 定位。

参考文献

[1]葛立. 基于双目视觉的六自由度机械臂手势识别控制系统研究[D].山东科技大学,2018.DOI:10.27275/d.cnki.gsdku.2018.000534.
[2]徐军, 刘春花, 孟月霞,等. 可穿戴手势识别控制器[J]. 电子技术应用, 2016, 42(007):68-71.
[3] Amma C , Krings T , B?Er J , et al. Advancing Muscle-Computer Interfaces with High-Density Electromyography[C]// ACM. ACM, 2015:929-938.
[4] Hackenberg G , Mccall R , Broll W . Lightweight palm and finger tracking for real-time 3D gesture control[C]// 2011 IEEE Virtual Reality Conference. IEEE, 2011.
[5] Weng C , Li Y , Zhang M , et al. Robust Hand Posture Recognition Integrating Multi-cue Hand Tracking[J]. DBLP, 2010.
[6] Li P , Ling H , Xi L , et al. 3D Hand Pose Estimation Using Randomized Decision Forest with Segmentation Index Points[C]// 2015 IEEE International Conference on Computer Vision (ICCV). IEEE, 2015.
[7] L. Ge, Y. Cai, J. Weng and J. Yuan, "Hand PointNet: 3D Hand Pose Estimation Using Point Sets," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 8417-8426, doi: 10.1109/CVPR.2018.00878.
[8] L. Ge, H. Liang, J. Yuan and D. Thalmann, "3D Convolutional Neural Networks for Efficient and Robust Hand Pose Estimation from Single Depth Images," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5679-5688, doi: 10.1109/CVPR.2017.602.
[9]Data-Driven Grasp Synthesis-A Survey[J].IEEE Transactions on Robotics: A publication of the IEEE Robotics & Automation Society.2014,30(2).289-309.
[10] 皮思远. 基于深度学习的工业机械手抓取检测方法研究[D].华南理工大学,2018.
[11] 何云龙. 基于双目立体视觉的手势识别技术[D]. 湖北:华中科技大学,2013. DOI:10.7666/d.D411617.
[12]赵君. 基于关键点检测与实例分割的手势交互算法研究[D].内蒙古科技大学,2020.DOI:10.27724/d.cnki.gnmgk.2020.000767.
[13]金珊杉. 基于点云深度卷积网络的三维手势姿态估计方法研究[D].辽宁大学,2020.DOI:10.27209/d.cnki.glniu.2020.000521.
[14]D. Tang, H. J. Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3D articulated hand posture. In CVPR, 2014.
[15] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. In TOG, 2014.
[16] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In CVPR, 2015.
[17]D. Kingma and J. Ba. Adam: A method for stochastic opti- mization. In ICLR , 2015.
[18] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algo- rithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software , 3(3):209– 226, 1977.
[19]王亚平,郭敏.非接触式激光测量点云数据预处理[J].上海计量测试,2006(02):27-30.
[20]C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In CVPR , 2017.
[21]M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks.

In NIPS , 2015.
[22] J. M. Van Verth and L. M. Bishop. Essential mathematics for games and interactive applications . CRC Press, 2015.
[23]马利,金珊杉,牛斌. 基于改进 PointNet 网络的三维手姿估计方法[J]. 计算机应用研究,2020,37(10):3188-3192. DOI:10.19734/j.issn.1001-3695.2019.06.0171.

附录

PointNet 清单：

train.py	PointNet 神经网络模型训练
utils.py	模型训练工具类
network.py	模型测试
dataset.py	参数配置
netR_59.pth	PointNet 网络训练结果存储
pretrained_net.pth	预处理网络训练结果存储
results	存储测试结果

上位机程序清单：

pycam.py	手势识别模块
hand_position_transform.m	仿生机械手臂运动算法模型
realsense_handtrack.ipynb	神经网络部署、可视化界面管理
start.py	系统启动文件
Uart_trans.py	串口通信函数文件
wifi_control.py	TCP 数据传输函数文件
paddle_control.py	踏板远程控制平台移动文件

仿生机械手臂控制程序清单：

startup_stm32f10x_hd.s	启动文件
main.c	主文件
ESP8266	esp8266 模块驱动文件
SysTick	系统定时器
robot.c	通信协议函数封装
stm32f10x_it.c	中断服务函数
uart	串口通信模块

作业平台程序清单：

cloudSTM32	外接摄像头驱动项目文件
paddle_control_car	远程接收踏板控制项目文件
CarMove	平台移动项目文件
delta.launch	激光雷达设置文件
hector.launch	SLAM 建模配置文件
hector_demo.launch	SLAM 建模实现文件
mbot_bringup.launch	获取激光雷达数据文件
cloud.py	外接摄像头远程控制文件