**分类号：TP391 U D C：D10621-408-(2023)4759-0**

**密 级：公 开 编 号：2019075067**

**成都信息工程大学**

**学位论文**

**基于三维点云数据的目标检测算法软件系统设计与实现**

|  |  |
| --- | --- |
| **论文作者姓名：** | **任宏培** |
| **申请学位专业：** | **机器人工程** |
| **申请学位类别：** | **工学学士** |
| **指导教师姓名（职称）：** | **王伟（讲师）** |
| **论文提交日期：** | **2023年5月30日** |

# **基于三维点云数据的目标检测算法软件系统设计与实现**

**摘要：**近几年来，自动驾驶技术逐渐成为学术界和工业界的研究的热点话题，引发了广泛的关注。在自动驾驶系统中检测到目标并精确感知目标的距离信息，这是保证无人驾驶安全性的关键。点云目标检测算法是近年来发展非常迅速的一个领域。早期的点云目标检测算法主要基于传统的计算机视觉技术，例如基于三维模型匹配，基于机器学习的特征提取等方法。但是这些方法受到点云数据本身的局限性，算法的鲁棒性和准确度都有限。目前，基于深度学习的点云目标检测算法已经得到了广泛应用，越来越多的3D数据被广泛应用于工业制造、物联网、自动驾驶等领域，而且研究者们对点云目标检测算法的研究还在不断深入，未来有望实现更高水平的技术突破和应用创新，这将极大地推动自动驾驶、机器人、智能制造等领域的发展。因此如何让用户更加便捷地评估不同点云数据和模型，成为了一个亟待解决的工程问题。

本设计的主要目的是利用PyQt对软件界面进行设计，以实现数据集加载、模型更换、py文件加载以及数据格式的转换等多种功能。该软件可以接受不同类型的目标检测模型，并且可快速进行模型推理，这种基于软件界面的数据加载方式提升了模型评估效率的同时，也降低了推理难度。由于点云数据来源日益丰富，出现点云数据与待测试模型不兼容的现象也时有发生。为此，本软件提供了点云数据格式转换的功能，使得数据更加兼容，同时也提高了数据处理的效率。开发者可以先利用本软件对数据进行格式转换，再对转换后的点云数据进行推理，从而极大地提高了推理效率。

本设计利于算法评估者快速地对算法的优缺点进行评估。经过对不同的模型进行评估实验，评估一个模型大概需要2-3分钟，这极大地提高了算法评估者的工作效率。

**关键词：**3D视觉；点云；UI设计；目标检测

**Design and Implementation of Target Detection Algorithm Software System Based on 3D Point Cloud Data.**

**Abstract:**In recent years, autonomous driving technology has gradually become a hot topic in both academia and industry, attracting widespread attention. Detecting and accurately perceiving the distance information of targets in autonomous driving systems is critical to ensuring safety. Point cloud object detection algorithms have developed rapidly in recent years. Early point cloud object detection algorithms were mainly based on traditional computer vision techniques, such as 3D model matching and feature extraction based on machine learning. However, these methods were limited by the inherent limitations of point cloud data, resulting in limited algorithm robustness and accuracy. At present, deep learning-based point cloud object detection algorithms have been widely used. More and more 3D data is being widely used in industrial manufacturing, the Internet of Things, autonomous driving, and other fields. Moreover, researchers continue to delve deeper into the study of point cloud object detection algorithms, which will eventually achieve higher levels of technological breakthroughs and application innovations. This will greatly promote the development of fields such as autonomous driving, robotics, and intelligent manufacturing. Therefore, how to enable users to more conveniently evaluate different point cloud data and models has become an urgent engineering problem.

The main purpose of this design is to use PyQt to design the software interface, implementing various functions such as data set loading, model replacement, py file loading, and data format conversion. The software can accept different types of target detection models and can perform rapid model inference. This data loading method based on the software interface enhances the efficiency of model evaluation while reducing the difficulty of inference. As the source of point cloud data becomes increasingly abundant, the phenomena of incompatible point cloud data with the target models sometimes occur. Therefore, the software provides a point cloud data format conversion function, making the data more compatible while also increasing data processing efficiency. Developers can first use the software to convert the data format and then perform inference on the converted point cloud data, significantly improving inference efficiency.

This design facilitates algorithm evaluators to quickly evaluate the pros and cons of algorithms. An evaluation experiment of a model generally takes 2-3 minutes, greatly improving the efficiency of the algorithm evaluator’s work.

**Key words:**3D vision;point cloud;UI design;target detection

目 录

论文总页数：36页

[1 引 言 1](#_Toc18955)

[1.1选题背景 1](#_Toc14241)

[1.2 国内外研究现状 1](#_Toc11968)

[1.3 研究目的和意义 3](#_Toc5228)

[1.4 章节安排 3](#_Toc8769)

[2 系统总体方案设计 5](#_Toc7268)

[2.1 主要设计功能及指标 5](#_Toc25225)

[2.2 总体技术路线 5](#_Toc21539)

[2.3本章小结 6](#_Toc20006)

[3 基于点云数据的目标检测的相关理论知识介绍 7](#_Toc25305)

[3.1 点云数据的采集 7](#_Toc3842)

[3.2 点云数据的格式 7](#_Toc13422)

[3.3 点云数据处理处理的方法 8](#_Toc19552)

[3.4 基于三维点云数据的目标检测算法 11](#_Toc4216)

[3.5 本章小结 13](#_Toc18090)

[4 环境搭建 14](#_Toc28245)

[4.1 软件环境的配置 14](#_Toc21696)

[4.2 软件的框架 14](#_Toc5289)

[4.3 可视化包Open3D 15](#_Toc6428)

[4.4 本章小结 15](#_Toc9773)

[5 软件界面的设计 16](#_Toc15086)

[5.1 工具的选择 16](#_Toc26429)

[5.2 登录界面的设计 16](#_Toc1867)

[5.3 软件界面的设计 17](#_Toc18974)

[5.4 本章小结 20](#_Toc8928)

[6 软件系统的测试 21](#_Toc22108)

[6.1 软件系统的测试 21](#_Toc24142)

[6.2 测试的结果分析 24](#_Toc3124)

[6.3 本章小结 24](#_Toc27261)

[结 论 25](#_Toc11298)

[参考文献 26](#_Toc21602)

[致 谢 28](#_Toc19467)

[声 明 29](#_Toc11269)

[附 录 30](#_Toc11938)

# 1 引 言

## 1.1选题背景

机器视觉是机器赖于对真实世界的感知与理解，是人工智能发展的一个新的方向，该技术使计算机使用传感器来取代人类的视觉感知能力，以便更好地了解周围环境的信息[1]，利用计算机技术，模拟人类大脑如何处理和理解外部环境的信息，有利于计算机系统做出正确的决策。近年来，随着人工智能技术的飞速发展，许多前沿科学技术，如无人机、智能家居、自动驾驶和移动机器人，已经被广泛应用于日常生活，比如，可以利用它们来识别室内环境的目标，并且可以操纵家庭服务机器人，为用户提供更加便捷的服务[2]；通过引入道路和车辆识别技术，我们可以在智能交通领域实现自动驾驶[3]。因此，作为机器感受外界的媒介，机器视觉技术在人工智能的研究中有着举足轻重的地位。

当前，计算机视觉技术已经发展到了一个新的阶段，它不仅涉及到目标识别、运动目标追踪和图像语义分割，而且这些技术在各个领域都有着重要的作用，尤其是在目标识别检测方面，它的研究价值更加突出。三维点云相比二维图像具有受光照、视点变化影响小,以及不存在投影变换问题等优势[4]，所以在机器人和自动驾驶领域，三维点云处理的研究越来越受到研究人员的关注。激光雷达是采集三维点云的理想传感器，具有极高的分辨率和精度，可以有效地收集三维点云数据，它能够提供更为准确、稳定的环境信息，并且能够抵抗外部条件，使其应用领域变得更为广泛。随着传感器技术的发展，点云数据可以提供丰富的场景及对象信息，现已成为自动驾驶、虚拟现实、机器人导航等应用的首选研究对象[5]，三维激光点云的目标识别技术也得到了蓬勃的发展，深度学习在2D图像等结构化数据处理中表现出了优越性能，对非结构化的点云数据分析处理已经成为计算机图形学的重要研究方向[6]。也正因为越来越多的研究人员加入对点云数据研究的大军中来，不同的模型也层出不穷，所以如何让用户更加便捷地评估不同点云数据和模型，成为了一个亟待解决的工程问题。

## 1.2 国内外研究现状

关于该领域类的研究现状主要分为以下两点，点云数据处理的研究现状和点云数据处理软件的研究现状：

1、点云数据处理的研究现状：

通过激光雷达传感器获取的点云数据可以提供大量的有用信息，其中包括但不限于地表点云、与目标检测无关的环境因素以及其他可能影响检测结果的噪声。在大多数情况下，干扰信息的比例可能超过点云数据的一半，这种情况不仅会严重影响目标识别的准确性，而且还会极大地增加计算量、提高效率，从而降低目标识别的成功率。随着技术的发展，激光雷达在测量外界环境信息的过程中，受到器件精度、振幅、散射等因素的影响，所收集的点云数据往往是杂乱无章的，甚至与实际情况不符，这种情况就被称为噪声点。在利用激光点云数据进行目标识别之前，必须采取一系列措施，包括仔细筛选和消除任何可能影响识别结果的噪音，同时确保有效的信息得到充分利用，这一步骤就是点云数据的预处理技术。 通过对点云数据进行预处理，可以最大限度地缩小其体积，同时保留其中的重要信息，从而使其具备良好的流畅性，从而为后续的点云分析提供了良好的基础，从而大大提高了目标识别的效率。因此，点云数据的预处理技术至关重要，其中包括：采用点云滤波器进行噪声抑制、精简信息、进行地表点划分。

在使用激光雷达进行扫描并收集三维点云数据的过程中，由于振动、实验设备和人为因素的影响，难免会出现噪音。随着场景的变得更加复杂，数据量也会增加，这将导致更多的噪音。随着技术的发展，点云滤波去噪已经成为一项至关重要的任务，它能够有效地抑制大量噪声，从而提高点云数据的质量和稳定性。因此，国内外学者纷纷投入资源，开展了大量的点云滤波去噪研究，以期达到良好的效果。

随着聚类算法的发展，刘大峰等人提出的一种基于聚类的核估计鲁棒滤波算法[7]已经成功地解决了点云去噪的问题，该算法能够有效地抑制噪声，同时也能够解决离群点的问题。然而，由于核函数的收敛过程十分复杂，使得该算法的实施变得十分困难，因此，在实际应用中，我们需要采取更加灵活的方法，才能够达到最佳的点云去噪效果；钟志鹏等人提出的OPTICS聚类和改良的双边滤波技术，可以有效地抑制大尺度噪声，同时也可以抑制小尺度噪声[8]，将这两种技术有机地结合在一起，可以获得更优的点云去噪效果[9]平滑算法在处理噪声点云方面表现出色，例如最小二乘法和曲率法。Fleishman和Jones等人提出的方法主要通过双边滤波来去除三维噪声网格[10]。赵灿和其他研究人员提出了一种基于最小二乘法的二次曲面拟合方法，用于消除噪声点，并取得了令人满意的结果[11]。张毅、葛宝臻及其他学者利用K-近邻点云的高斯核函数，开发出一种全新的点云去噪算法[12]，它不仅具备良好的噪声抑制功能，还具备局部曲率特征、平面投影及双边滤波的优势，使得点云的分类精度大大提升[13]，为数据处理带来更多便利。曹爽、刘强等人提出的双边滤波算法具有显著的优势，首先，它采用了特征点选择，有效抑制了噪声干扰，同时也能够有效地保留特征[14]；其次，它采用了通过迭代计算获取的法向量，其实验结果表明，它的性能优异，可以应用于实际[15]场景中。

综上所述，点云数据的预处理技术有着多种多样的选择，根据不同的应用环境，采取相应的技术手段，并将这些技术结合起来，从而获得更加优质的处理结果。

2、点云数据处理软件的研究现状:

PCL是一款免费且开源的点云数据处理库，由斯坦福大学的研究人员和其他社区贡献者共同开发，用C++编写。它包含了大量的点云数据处理算法和应用，例如点云数据的滤波、分割、配准、识别、重建等，可以处理多种点云数据格式，如PCD、PLY、OBJ等，并支持ROS（机器人操作系统）平台。

目前，比较成熟的三维点云数据处理软件和模块也有很多[16]。在国外：CloudCompare是一款免费且开源的点云数据处理软件，由法国土木工程师Daniel Girardeau-Montaut开发。它可以用于点云数据的可视化、滤波、重采样、修补等处理，还可以进行点云的比较、配准、分割、分类、模型重建等应用，支持多种点云数据格式，如PLY、OBJ、LAS等。CloudCompare还提供了一个在线版本，称为CloudCompare Online。Open3D是一款免费且开源的点云数据处理库，由亚利桑那州立大学、清华大学、英伟达公司等单位共同开发，用C++和Python两种语言编写。它支持点云数据的滤波、配准、分割、重建、可视化等处理，还提供了一些机器学习相关算法，如深度学习模型的训练，支持多种点云数据格式，如PCD、PLY、OBJ等，并提供Python接口。MeshLab是一款免费且开源的三维网格处理软件，由意大利的独立开发者Paolo Cignoni开发。它支持点云数据的导入和导出，可以进行点云的滤波、配准、分割、重构等应用，同时也支持三维网格数据的处理，例如网格的平滑、剖分、退化等，支持多种网格数据格式，如OBJ、PLY、STL等。Geomagic是一款商业软件，由3D系统公司开发。它包含了一套完整的3D数字化解决方案，用于产业设计、制造、医疗等领域，支持点云数据的处理、分析、编辑、配准等应用，可以进行三维重构、CAD分析、快速原型制作等任务，支持多种点云数据格式，如PTS、PTX、LAS等。CloudCompare Online是CloudCompare的在线版本，可以直接在浏览器中使用。它支持大多数CloudCompare的功能，包括点云数据的可视化、滤波、重采样、修补、比较、重建等处理，支持多种点云数据格式，如PLY、OBJ、LAS等。它还支持将点云数据导出为多种格式，并提供了一些机器学习相关算法，如深度学习模型的训练。

尽管三维点云数据处理软件种类繁多，但它们仍有一定的局限性，尤其是在点云数据处理领域，如：智能化水平偏低、需要大量人力介入、集成度不高等。

## 1.3 研究目的和意义

本课题的主要工作是利用PyQt5编写软件界面，将各种参数的配置方式集成在软件界面上，对系统进行优化和美化，这有利于算法评估者快速的对算法的优缺点进行评估。

## 1.4 章节安排

第一章：介绍课题的研究背景，然后讲述了国内外的研究现状，最后提出本课题的主要内容。

第二章：整个系统的介绍，涉及到系统完成的性能指标、系统的总体设计以及系统各个功能模块。

第三章：介绍了基本的点云数据的处理方式和各种处理方式后的期望目标，以及各种基于三维点云数据的目标检测算法。

第四章：介绍了软件环境的搭建以及各种实现软件功能工具的选择缘由。

第五章：介绍了软件界面编写，以及编写工具的选择，以及对整个系统进行了包装和美化。

第六章：介绍了软件系统的测试结果，主要测试分为接纳性测试和速度测试。

# 2 系统总体方案设计

## 2.1 主要设计功能及指标

本课题主旨在设计一款目标检测算法软件系统，具备模型的选择，数据的修改和结果的输出等功能。其具体任务和要求如下:

(1)数据采集：点云数据主要来源于国际开源的数据和指导老师提供的企业数据；

(2)数据处理：不同格式的点云数据转化为mmdetion3d框架可接受的格式；

(3)特征提取：点云数据的体素化和几何特征的提取；

(4)目标检测算法的选择与实现：学习并熟悉不同的3d点云目标检测算法；

(5)界面设计与实现：通过Open3D+PyQt的学习，实现友好界面。

## 2.2 总体技术路线

主要通过Open3D+PyQt对软件界面进行设计，通过软件上的功能模块，让用户能够自主选择使用的模型和所需要的数据，软件后续会将所得到的结果进行输出，方便用户对模型进行快速的评估具体流程图如图2.1所示，还具备点云数据格式转换等功能。

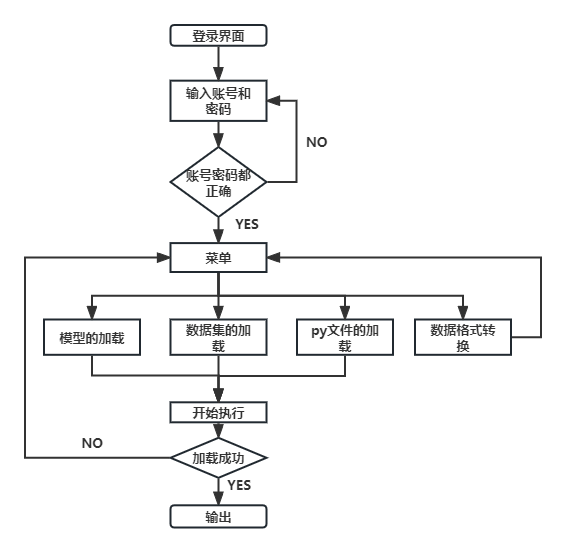


图2.1 系统流程图

## 2.3本章小结

本章重点讲述了此系统完成的功能指标，以及此系统的整体框架，通过对本章节的阅读，可以对此系统的整体设计和工作流程有一个大致的了解，并且因为此系统采用了模块化设计，也能够对此系统的各部分功能模块有一个初步的了解，后续将对各部分功能模块进行详细讲述。

# 3 基于点云数据的目标检测的相关理论知识介绍

## 3.1 点云数据的采集

点云数据是通过激光雷达、摄像头或其他传感器采集的。以下是一些常见的点云数据获取方式：

激光雷达：激光雷达通过发射激光束来扫描环境，并通过接收返回的激光束来测量环境中物体的距离和形状。激光雷达可以在短时间内获取大量的点云数据，因此是常用的点云数据获取方式。

摄像头：摄像头可以通过捕捉连续的图像来获取点云数据。通过在连续的图像之间进行视差计算，可以确定物体的距离，并将其转换为点云数据。

其他传感器：除了激光雷达和摄像头之外，还有其他类型的传感器可用于获取点云数据。例如，声纳传感器可以在水下环境中获取点云数据，而雷达传感器则可以在弱光环境下获取点云数据。

无论使用哪种传感器，获取到的点云数据通常需要进行后处理和滤波，以去除无效或噪声数据，并提取有用的信息。

## 3.2 点云数据的格式

由于点云数据的来源十分的丰富，所以在读入数据的时候，首先得把不同点云数据的格式转换为软件系统可接受的格式。

常见点云存储方式有pcd、ply、txt、bin文件。 pcd点云格式是pcl库种常常使用的点云文件格式。一个pcd文件通常由两部分组成：分别是文件说明和点云数据。

PCD数据格式是点云库（PCL）的一种特殊的文档形式，它可以根据点云库的需求进行灵活的定制，并且可以支持n维点的扩展，从而提升点云库的处理效率。PCD文档由一个头文件以及一个数据部分组成，前者可以提供点云的完整概览，而后者则可以提供点云的细节特征，以便进一步深入研究点云的特征。如版本、字段、大小、类型、数量、宽度、高度等，数据部分为点的坐标和属性，由点的笛卡尔坐标构成，文本模式下以空格做分隔符。文件格式有文本(ASCII)和二进制(Binary)两种存储格式。pcd格式使用ASCII码进行编码，包含一个头文件；而bin仅仅就是数字的二进制拼接。因此pcd到bin的转换就是把每一行的数字（代表一个点）拼接在一起，反之亦然。

ply转bin的过程就是读取ply中的点云数据为numpy矩阵形式，然后直接保存为bin格式就可以了。因此，ply格式读取并不一定需要plyfile库，也可以是之前介绍的open3d库，或者不适用任何库，直接用with open打开文件读取即可。

ASC、XYZ与TXT都是点云格式中一种常见的文本格式，可以用于存储点云数据以及其他3D几何数据。在这些格式中，每个点占据一行。每个点在一行中以x、y、z三个坐标值的顺序给出，每个坐标值之间用空格、制表符或逗号进行分隔。此外，可以在每个点的坐标值之后加上可选的点属性信息，这些属性信息也以空格、制表符或逗号分隔。文本格式的文件可以使用任何文本编辑器打开，具有易于阅读和编辑的优点。

LiDAR（LiDAR））是一种用于记录和传输高精度数字图像的工业标准格式，它以二进制形式存储，并且由美国摄影测量与遥感学会2003年发布。LAS文件中包含了多种重要的信息，如三维坐标、多次回波、强度、扫描角度、分类、飞行航带、飞行姿态、项目、GPS以及数据点的颜色等。LAS致力于建立一个兼容性强、功能齐全的数据库，以便各种硬件和软件厂商能够以一致的、可扩展的方式实现数据的存储与共享。

## 3.3 点云数据处理处理的方法

点云作为一种三维环境数据因其具有较高的精度一直被广泛关注并应用于多种场景任务之中[17]。点云数据转化完毕之后，在加载之前，还得进行一定的预处理。在激光雷达点云数据中，由于大部分数据是地面点数据，而且这些地面点数据呈现出纹理状，因此，在进行目标物体分割时，必须先进行过滤，以确保点与点之间的距离接近，以便更准确地识别出障碍物点云。否则，分割算法将无法正常运行，从而影响最终的分割结果。因此，通过去除地面点云数据，可以有效地减少数据量，并且提升分割算法的准确性。为了更好地处理地表点云，许多学者开发了多种技术，其中包括基于栅格图的、基于三维激光雷达原始扫描线的、以及其他更加精细化的技术，它们能够更加稳健、可靠地处理地表点云。利用激光雷达技术获取的点云覆盖了绝大多数的地表特征；常用的栅格图方法地面滤除点云方法有栅格高度差法、法向量方法和高度法。

1. 点云的滤波

在对点云进行三维重建中,不可避免地会产生各种噪声[18]，点云滤波的目的是去除噪声点和异常点。常用的滤波方法包括基于体素的滤波、基于统计的滤波和基于距离的滤波。其中，基于体素的滤波将点云数据离散化为一组立方体，并通过统计每个立方体中包含有多少个点来进行降采样和去噪；基于统计的滤波则是通过计算点云数据的标准差来去除离群点；基于距离的滤波通过将点云数据中距离其他点较远的点删除。

1. 点云配准

点云配准通常用于将两个或多个点云进行对齐。配准技术是计算机视觉中一个重要的领域[19]，点云配准广泛应用于三维重建、参数评估、定位和姿态估计等领域,在自动驾驶、机器人和增强现实等新兴应用上也有点云配准技术的参与[20]。其中经典的方法包括ICP（最近点迭代），即通过将模型的点云向目标点云进行迭代匹配，最小化两个点云之间的距离。还有NDT（正态分布变换），该方法通过建立每个点的概率分布，寻找两个点云之间的刚性变换，使得点云之间的误差最小，最终实现对齐。以下是几种常用的点云配准算法：

* 1. 迭代最近点算法（ICP）

迭代最近点算法（Iterative Closest Point，ICP）是一种基本的点云配准算法，其基本思想是通过不断迭代找到两个点云中距离最近的点对，并对齐两个点云。具体而言，ICP算法首先预估一个初始的变换矩阵，然后在每次迭代中寻找对应的点对，并通过最小化这些点对之间的距离，来不断更新变换矩阵。当匹配误差小于预设阈值时，ICP算法终止迭代，输出最终的变换矩阵。

* 1. 高斯分布变换法（NDT）

高斯分布变换法（Normal Distribution Transform，NDT）是一种基于概率分布的点云配准算法。NDT算法通过计算点云的特征分布来快速匹配两个点云。具体而言，NDT算法首先在参考点云上定义一个体素网格，并分别计算每个体素的特征分布的均值和协方差矩阵；然后，对于目标点云中的每个点，计算其与参考点云中各个体素的贡献值，从而得到目标点云的特征分布。最后，通过计算参考点云和目标点云的特征分布之间的KL散度，来计算两个点云的相似度，并得到最终的配准结果。与ICP算法相比，NDT算法更适合处理稀疏、噪声干扰较大的点云数据。

* 1. 全局优化ICP算法（Go-ICP）

全局优化ICP算法（Global Optimization, ICP optimization，Go-ICP）是对ICP算法的改进，主要的思想是采用全局的优化方法将问题转化为一个全局最小化问题，通过优化变换矩阵，使得全局匹配误差最小。具体而言，Go-ICP算法首先对点云数据进行特征提取，例如表面法线、曲率等，然后利用这些特征来构建成本函数，对变换矩阵进行全局优化。

* 1. 广义迭代最近点算法（GICP）

广义迭代最近点算法（Generalized Iterative Closest Point，GICP）是对ICP算法的改进，主要是通过对相邻点进行加权，解决了单点的不能完全捕捉表面变化的问题。具体而言，相邻点的加权是通过计算矩阵的广义逆（Moore-Penrose伪逆）来实现的。GICP算法相比于ICP算法，能够在不改变初始化位置的情况下，获得更加准确的配准结果。因此，GICP算法适用于处理不同视角下的点云配准问题。

* 1. 基于高斯混合模型的稳健点匹配算法（RBG-ICP）

基于高斯混合模型的稳健点匹配算法（Robust Point Matching based on Gaussian Mixture Models，RBG-ICP）是一种近年来提出的点云配准算法，该算法通过建立高斯混合模型来对点云数据进行建模，从而提高了配准的鲁棒性和准确性。具体而言，RBG-ICP算法首先将点云数据转换为高斯混合模型，并通过混合成分之间的关系来计算每个点在模型上的高斯混合成分。然后，通过根据每个点在模型上的高斯混合成分来计算加权点间距离，最终得到配准结果。

3、点云分割与分类

点云分割（Point Cloud Segmentation）是将点云数据进行分割，将同一类物体的点划分为同一子集的过程。点云分类（Point Cloud Classification）则是将点云数据中的每个点标记为属于某种类别的过程。以下是几种常用的点云分割与分类方法：

* 1. 基于规则的方法

基于规则的方法需要人工设定若干分类和分割规则，例如根据点的颜色、密度、距离等特征将点云分割成不同的区域。这种方法的好处是容易理解，可解释性强，对于较为简单的场景能够快速实现。但是对于复杂场景，手动设计规则会面临困难，在实践中的应用受到较大限制。

* 1. 基于机器学习的方法

基于机器学习的方法需要构建一个模型，使用人工标注的数据来训练模型。常用的分类器如支持向量机（SVM）、随机森林、决策树等。这种方法利用机器模拟人类学习的过程，不需要手动设计规则，可以自动从数据中提取特征。但这种方法也需要大量的数据集来支撑，同时模型的泛化能力不如深度学习方法。

* 1. 基于深度学习的方法

基于深度学习的方法利用深度神经网络对点云进行语义分割和分类。点云处理方法主要有PointNet、PointNet++、KPConv、DGCNN等。使用深度学习的方法能够自动提取特征、对数据进行降维和分类，同时通过大规模训练可以学习到复杂的特征和在全局上对点云进行建模。但是由于模型的复杂性，需要投入大量的计算资源，还需要大规模数据集的支撑。

* 1. 基于图像处理的方法

基于图像处理的方法将点云转换成图像形式，应用图像处理技术进行分割和分类。典型的方法是利用深度图像，或者通过对点云投影形成二维图像[21]，然后应用现有的图像分类算法来进行分割和分类，例如二维卷积神经网络。这种方法采用了现有图像处理技术，能够直接处理像素点，适应性较强。但是，点云转换成图像后会出现信息的损失，如图像噪声和平面变换等问题，这对分类和分割的结果会产生一定影响。

4、目标识别检索

点云的目标识别检索是指利用计算机视觉技术，对三维点云数据中的目标进行自动化识别和检索，3D点云目标检索是计算机3D视觉中的一个关键技术[22]。点云数据通常是由多个相邻测量点所构成的集合，旨在呈现三维场景中的各个物体。

点云目标识别检索系统需要先进行处理，如点云去噪、点云配准、点云拼接等操作，然后使用机器学习、深度学习等算法进行目标识别。这些算法可以分为两类：基于特征描述子的方法和基于深度学习的方法[23]。特征描述子是对点云数据中的每个点进行数字化表示的方法，包括3D-SIFT、SHOT、SFH等；而基于深度学习的方法则是采用卷积神经网络等深度学习技术，利用大规模标注的点云数据训练网络模型，使其具备目标识别能力。

点云目标识别的应用场景包括自动驾驶、机器人控制、灾害救援等，旨在实现对空间中目标的自动识别和判断，从而进一步完成机器人掌控、飞行控制等智能化任务。此外，点云目标检索也可以用于对点云数据库中的目标进行快速检索，找到与查询点云相同或相似的目标点云，从而实现对三维场景中目标的快速相似性匹配。

5、点云重建

点云重建的目的是将散乱的点云数据拼接成一个完整的3D模型。常见的重建方法包括基于体素的重建、显式重建和基于隐式曲面重建。基于体素的重建是将点云数据离散化为一个个体素，通过统计体素内部的点的数量来构建模型，速度很快，但精度较低。显式重建（张成网格）则是将点云转换为表面上的一组多边形，常用于实时渲染。基于隐式曲面重建，则是使用隐式函数来描述点云数据的连续曲面性质，算法可以进行更为精确的表面重构。

## 3.4 基于三维点云数据的目标检测算法

目前，基于三维点云数据的目标检测算法主要包括以下几种：

基于视觉几何学的方法：该方法通过分析三维点云数据中的平面和边缘等特征，利用基于视觉几何学的技术实现目标检测和定位。该方法的优点是算法简单，但是对于复杂场景和非刚体物体的检测存在一定的限制。

基于深度学习的方法：该方法通过构建三维卷积神经网络（3D CNN），实现对三维点云数据的特征提取和目标检测。该方法的优点是具有较高的检测精度和鲁棒性，但是需要大量的数据和计算资源来训练和优化网络。

基于点云分割的方法：该方法将三维点云数据分割为不同的物体，然后对每个物体进行分类和定位。该方法的优点是可以有效地处理多物体和遮挡情况，但是对于小物体和低密度点云数据的检测存在一定的限制。

基于光流法的方法：该方法通过对相邻点云帧之间的运动进行分析，实现对目标物体的检测和跟踪。该方法的优点是具有较高的实时性和鲁棒性，但是对于场景的变化和摄像机姿态的变化存在一定的限制。

综上所述，基于三维点云数据的目标检测算法具有广泛的应用前景，不同的算法适用于不同的场景和需求。在实际应用中，需要根据具体情况选择合适的算法，并结合硬件设备和系统架构进行优化和实现。

1、PointNet与PointNet++

PointNet是一种端到端处理三维点云的神经网络模型，PointNet++是基于PointNet的改进版本[24]，主要改进了处理多尺度信息的能力。PointNet的设计是直接对点云进行处理，通过max pooling、MLP等操作提取特征，最后通过全连接层进行分类或回归任务。但是PointNet只能处理点云数据的全局信息，无法提取局部信息[25]。PointNet++通过引入一种金字塔式结构，针对不同尺度的点云进行处理，增强了处理多尺度信息的能力，可以处理更复杂的场景。

2、Frustum-PointNet

Frustum-PointNet是一种基于2D图像边界框的点云目标检测算法。其方法首先利用图像的2D目标检测模型提取目标2D区域[26]，然后根据2D检测框在点云中进行裁剪，在裁剪过的点云中使用PointNet对点云进行分类和定位。

3、VoxelNet

VoxelNet网络模型是第一个基于点云的端对端目标检测网络，只利用点云数据来生成高精度的3D目标检测框,具有十分良好的效果[27]。其主要贡献在于提出了一种大规模点云数据集 KITTI 数据集，从而推动了点云目标检测的发展。VoxelNet先将点云转换为体素，然后对每个体素进行处理，提取特征。与 Frustum-PointNet 类似，VoxelNet 也是通过 PointNet 网络进行分类和回归任务，2019CCDC提出了一种新的基于PointNet和VoxelNet的场景分割方法来提高语义分割的效率和准确性[28]。

4、ComplexYOLO

ComplexYOLO 是基于 YOLO2 的算法，利用自适应的体素池化使网络更具有处理小尺寸目标的能力，并提供了更高的检测精度。它可以快速地检测三维物体并进行分类和定位。相比于 Yolo2，ComplexYOLO 改进了特征提取和池化方法，使得 ComplexYOLO 可以快速、准确地检测小的物体。

5、MV3D

MV3D是一种基于融合点云和图像信息的算法，提出了三维空间感受野机制，可以增强对于球形物体处理的能力。MV3D 首先使用卷积神经网络处理图像，提取图像特征，然后使用 VoxelNet 对点云进行处理。在两个网络的输出层进行融合，同时进行分类和定位。由于使用了两种不同类型的数据，MV3D的准确率相对其他方法更高。

6、PointPillars

PointPillars是一种基于点云的目标检测算法，其主要思路是将点云空间进行划分，然后通过二维卷积神经网络（CNN）提取特征，最后使用后处理模块进行物体的识别和定位。

pointPillars的具体实现步骤如下：首先将点云数据划分为矩形格子，并将每个格子中的所有点归一化到相同的高度上，形成二维网格。这一步在PointPillars算法中被称为“Voxelization”（体素化）；然后采用二维卷积神经网络搭建多层特征提取网络，使用对应的边缘池化（edge pooling）和类别池化（class pooling）函数将每个格子的特征映射到固定的二维矩形区域上；紧接着，将池化之后的特征向量作为网络输出，包括边界框类别（分类）和边界框位置（回归）。这一步在PointPillars算法中被称为“Pillar Feature Net”。最后，使用非极大抑制（NMS）后处理步骤来提高检测精度和效率[29]，其中通过高置信度分数和低重叠率来决定哪些重复的物体框应该被保留。

PointPillars的主要优点在于它可以快速地处理大规模点云数据，并具有较高的检测精度。不少研究人员认为，PointPillars是一个重要的三维物体检测算法，在未来自动驾驶车辆和人工智能领域会得到广泛应用。大量实验表明，PointPillars在速度和准确性方面都大大优于以前的编码器[30]。

## 3.5 本章小结

本章主要介绍了目标检测技术在三维点云数据中的应用，涉及到了基于点云和基于图像和点云融合的目标检测算法。在基于点云的算法中，PointNet、VoxelNet和PointPillars是其中比较典型的算法，在点云数据的处理、特征提取和后处理方面都有所创新。而基于图像和点云融合的算法，如MV3D等，则更加注重多源信息的利用和如何提高模型性能的问题。同时，文章也指出了目标检测技术在应用中仍存在一些问题和挑战，如点云数据的噪声和稀疏性，物体形状的多样性等。未来的研究和应用方向主要是提高算法的鲁棒性、准确性和普适性，为自动驾驶、智能机器人和安防等领域的应用提供更好的支持。

# 4 环境搭建

通常，在软件开发领域中，环境搭建是重要的一步，在搭建完成环境之后，开发人员可以开始编写、测试和运行他们的代码。常见的环境搭建包括编程语言和框架的配置，数据库和缓存的安装。一旦环境搭建成功，开发人员就可以根据其需求进行编码、构建和部署。

## 4.1 软件环境的配置

如图4.1所示，环境搭建选择使用anaconda创建一个虚拟环境，使用python作为软件开发语言，使用PyTorch作为开发框架并通过CUDA编程利用GPU对程序的运行进行加速，通过MIM包管理工具下载安装MMDetection3D，并安装可视化工具Open3D，并且使用PyQt对软件界面进行设计。

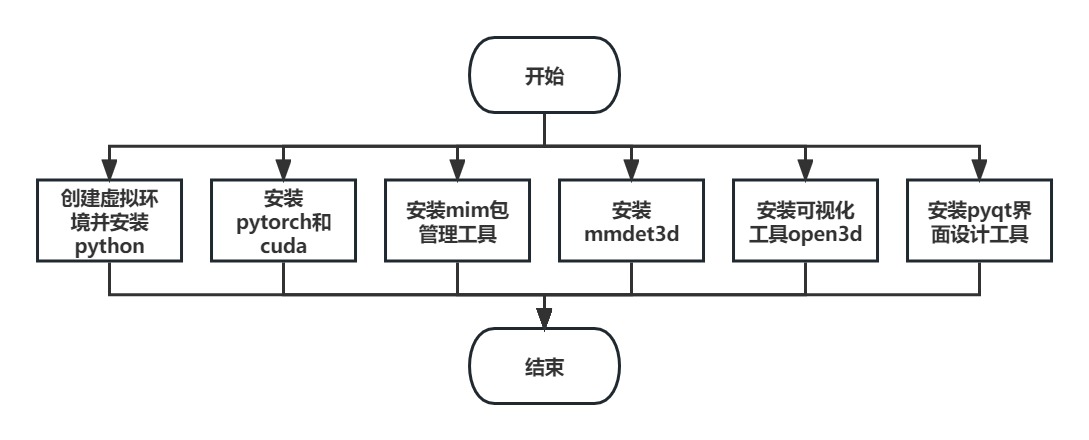


图4.1 环境搭建

## 4.2 软件的框架

本软件基于MMDetection3D的框架，MMDetection3D是基于PyTorch 3D 深度学习框架的目标检测和物体检测库，主要针对三维点云数据进行物体检测和分割任务。MMDetection3D在MMDetection库的基础上，将二维目标检测框架拓展到三维点云数据，并支持了很多最新的3D目标检测和分割算法，如Part-A2 Detectors, PV-RCNN, PointRCNN, 3DIOU Loss等。这个库主要功能包括：

* 1. 支持点云数据上的常见检测任务，如目标检测和分割。
  2. 工具库包含数据集准备、数据增强、3D可视化等模块。
  3. 支持多个3D检测网络，参考经典工作和最新研究，包括PointRCNN, PV-RCNN, SECOND, PartA2等算法。

根据不同的任务提供了对应的评估和工具函数，如检测AP，分割IoU等。

支持单卡和多卡训练，提供具有批次导入优化的分布式训练。MMDetection3D是一个开源且易于使用的目标检测库，它提供了一系列常见的3D点云数据任务，可以低成本地建立检测模型。因为其支持多种常见的3D目标检测算法，因此可以在解决实际问题时选择最适合的算法。

本软件主要是在MMdetion3D的框架下，集成模型的更换和数据集的选择等功能，方便算法评估者对模型和数据集进行快速的评估。

## 4.3 可视化包Open3D

Open3D是一个开源的、跨平台的3D数据处理库，其主要目的是简化3D数据处理和机器学习领域中的基础算法。Open3D支持大量的3D数据处理任务，包括点云和网格处理、3D可视化、图像处理、相机姿态估计、物体检测等。它主要提供了以下功能：

* 1. Point Cloud (点云)和Triangle Mesh (三角形网格)数据类型。
  2. I/O读写：Open3D支持大量的3D数据格式的导入和导出，如PLY、OBJ、OFF、3D Max等。
  3. Registration (配准)：Open3D提供了ICP，RANSAC等配准算法的API，用于点云、网格和深度图像等数据类型的配准。
  4. Reconstruction (重建)：Open3D支持从多个视角重建3D几何形状，从RGB图像或深度图像到三角形网格等数据类型的重建。
  5. Visualization (可视化)：Open3D提供了3D数据类型的可视化接口，支持交互式3D可视化，包括相机漫游、模型旋转、剖切等功能。
  6. Object Detection (目标检测)：Open3D引入了一些经典的物体检测算法来进行3D目标检测，如基于点云的3D目标检测(PointNet)、MV3D等。
  7. Learning (机器学习)：Open3D提供了一些基础的机器学习模型和工具函数，如支持向量机、最近邻算法和神经网络。它还允许使用TensorFlow或PyTorch等深度学习框架进行更复杂的学习任务。

综上所述，Open3D是一个功能丰富、易用性好的3D数据处理库，提供各种基础算法和工具函数，能够快速开发和构建应用程序。它已经成为了3D数据处理、机器学习、计算机图形学等领域的重要工具，所以选择Open3D作为可视化工具。

## 4.4 本章小结

本章主要讲述了软件开发环境的搭建，如安装Anaconda，在其中配置Python开发环境，并安装PyTorch和MMDetection3D等相关依赖库。同时，还需要安装CUDA等GPU加速库，以提高算法的训练和测试速度，以及可视化工具的Open3D的选择。这样更方便算法评估者对模型进行快速的评估。

# 5 软件界面的设计

软件的功能在上文中已经得到了基本的体现，下述内容主要为软件界面的设计，作为一款算法评估的平台，一直通过在编辑界面对程序的参数进行设置让使用者用起来十分的不方便，为了方便算法评估者的工作，所以决定制作一个界面，将算法评估者经常使用到的功能集成到软件界面上。

## 5.1 工具的选择

PyQt是使用Python语言和Qt应用程序框架编写的一组Python模块，用于创建自定义GUI应用程序。它是Qt C++应用程序框架的Python绑定，可以帮助程序员使用Python编写高效且功能强大的桌面应用程序。PyQt主要分为两个组件模块：Qt模块和Python模块。Qt模块包括许多Qt库，为Python提供了一个非常强大的GUI编程平台。PyQt具有许多优点，包括跨平台，易于使用，灵活性高，引入了Qt的编程技术等等。在使用PyQt编写GUI应用程序时，可以充分利用Qt提供的广泛且广泛的GUI组件集，如窗口、按钮、文本框、标签、滑块、进度条、列表、树、日期控件等等。Python程序员可以使用PyQt开发GUI（图形用户界面）应用程序。与其他Python GUI库相比，PyQt使程序员可以使用C++应用程序框架进行操作，可以获得更好的性能和效率。

总的来说PyQt是一个高效、简单、灵活的框架，具有很高的可扩展性和跨平台性。它对Python开发者来说是非常有用的，可以用于开发各种GUI应用程序。所以选择PyQt来对软件GUI界面进行设计。

## 5.2 登录界面的设计

在软件界面设计之前，先设计了一个登录的界面，设计登录界面的主要目的是确保应用程序或网站的安全性，控制用户的访问权限，保护用户的个人信息，以及为用户提供方便快速的登录流程，如图5.1所示。



图5.1 登录界面

如若登录密码错误，会要求用户重新输入密码，这样对用户的隐私得到了较好的保护，具体步骤如图5.2所示。

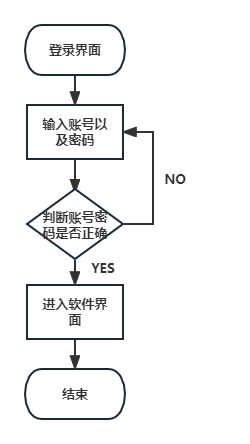


图5.2 登录界面流程图

## 5.3 软件界面的设计

软件主要在菜单栏下实现了了数据集的加载、模型的更换、py文件的选择还有退出四个功能，因此对应设计了一个菜单栏四个按钮，由于使用的是Open3D的可视化包，会单独弹出窗口，因此在软件界面上设计了一个开始运行的按钮，在数据集、模型以及py文件都加载好了之后，点击开始执行，软件将调用demo对数据集和模型进行评估，若三者之中少任何一项软件都将不会执行下一步的程序。如图5.3所示。

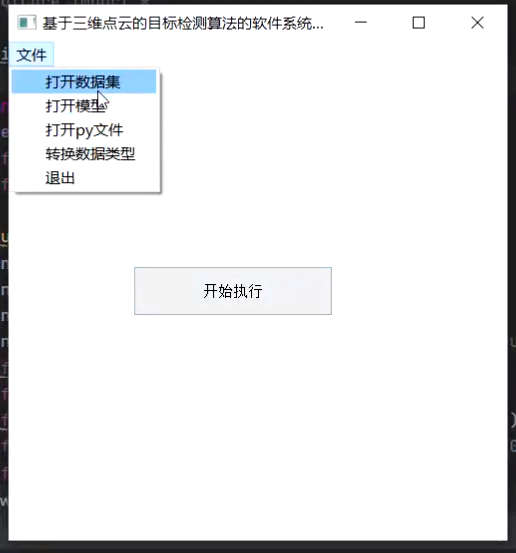


图5.3 软件界面

点击打开数据集的按钮，会出现图4.4的菜单，选择好需要加载的数据之后，点击打开，会出现图4.5的文字输出，表示数据加载成功。

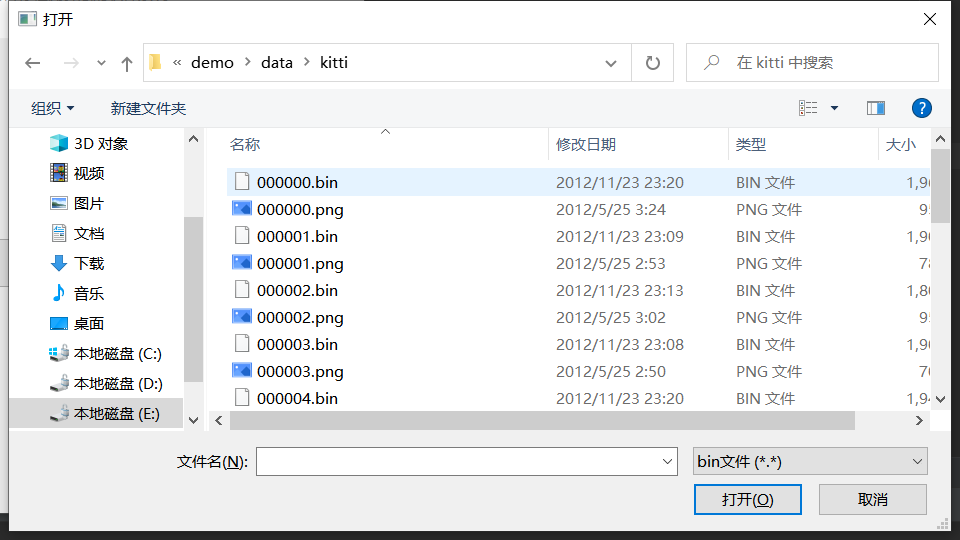


图5.4 选择数据集



图5.5 数据加载成功

数据加载成功之后，点击打开模型的按钮，会出现图4.6的菜单，选择好需要加载的模型后，点击打开，会出现图4.7的文字输出，表示模型加载成功。

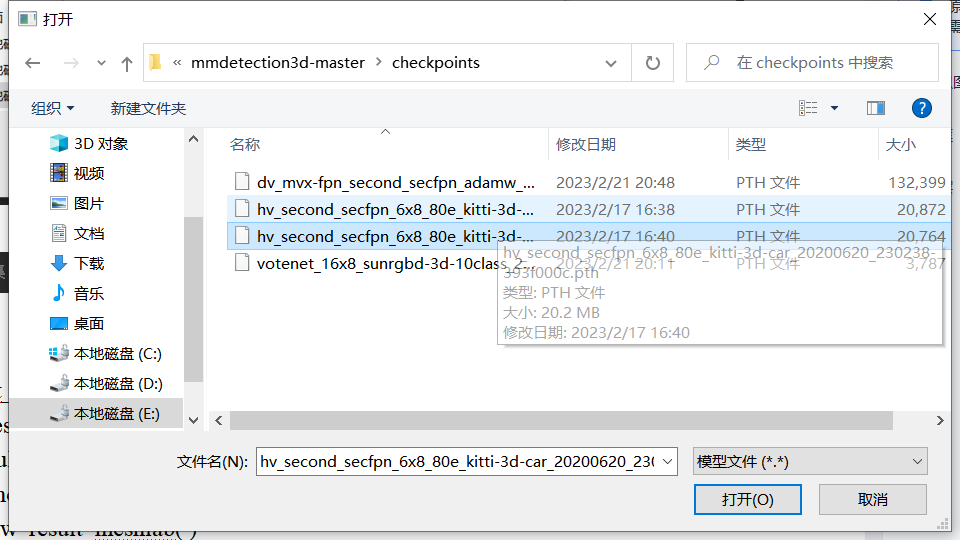


图5.6 选择模型



图5.7 模型加载成功

模型和数据集加载完毕之后，点击打开py文件的按钮，会出现如图4.8所示的菜单，选择与模型相对应的py文件，然后点击打开，选择成功后会出现如图4.9的文字输出。

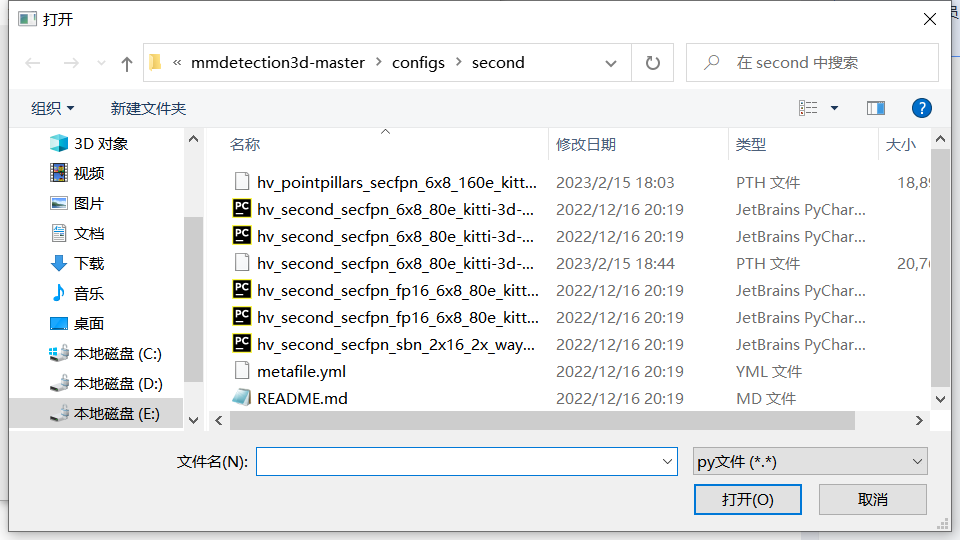


图5.8 模型对应的py文件



图5.9 py文件加载成功

上述所需文件加载完毕之后，点击开始执行的按钮，即将对加模型进行评估，如果需要更换数据集或者模型，可重复以上操作，软件将对前面选择的模型和数据集进行覆盖。

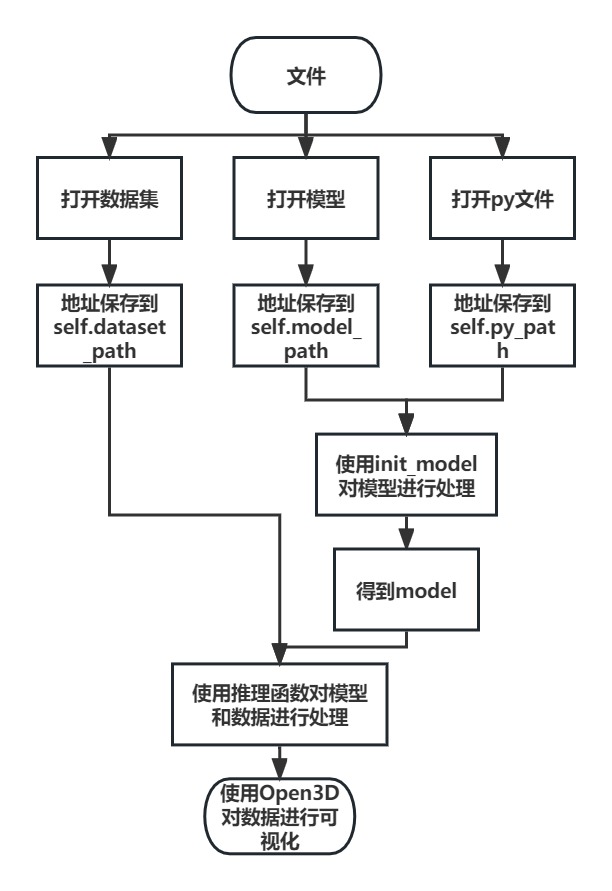


图5.10 软件的内部逻辑

图5.10是软件功能实现的重要部分，操作时打开的py文件将存储到self.py\_path中，打开的数据集文件将保存到self.dataset\_path中，打开的模型文件将保存到self.model\_path，然后通过调用init\_model函数和inference\_detector函数对参数进行解析，然后通过Open3D包中的 show\_result\_meshlab函数对点云数据的检测结果进行输出。

## 5.4 本章小结

本章主要讲述了软件登录界面的设计，GUI界面的设计以及软件功能的实现，主要是通过PyQt对软件需要的按钮进行了布局以及按钮事件的绑定，然后通过connect函数连接起来，基本上没有遇到太大的问题，本章的工作较为顺遂，至此，软件的环境搭建，界面以及功能模块已经全部完成，剩下的将是对软件系统的优化以及软件功能进行测试。

# 6 软件系统的测试

本软件主要包括两个方面，一方面是软件功能的实现，二是软件GUI界面的设计，至此，软件的基本设计已经完成，接下来将使用软件对不同的模型进行评估，测试其模型的准确度。

## 6.1 软件系统的测试

HV-SECOND是一个基于双视点深度图（HSVs）的3D目标检测器，是基于SECOND模型的改进。它结合了双目立体视觉和深度学习技术，在KITTI数据集上实现了非常优秀的3D车辆检测效果。

VoteNet是一种基于深度学习的3D目标检测模型，能够准确地检测3D场景中的物体。它的主要特点是采用了投票机制和集成学习来提高检测准确度，并且适用于不同类型的点云数据。该模型在sunrgbd数据集上进行了训练，并且可以检测出许多常见的3D物体。

在本次测试中，先采用HV-SECOND车辆识别模型，测试其在不同场景下的目标检测精度；后采用VoteNet模型进行室内家具的识别，并记录模型评估的时间。



图6.1 点云数据的二维场景图片

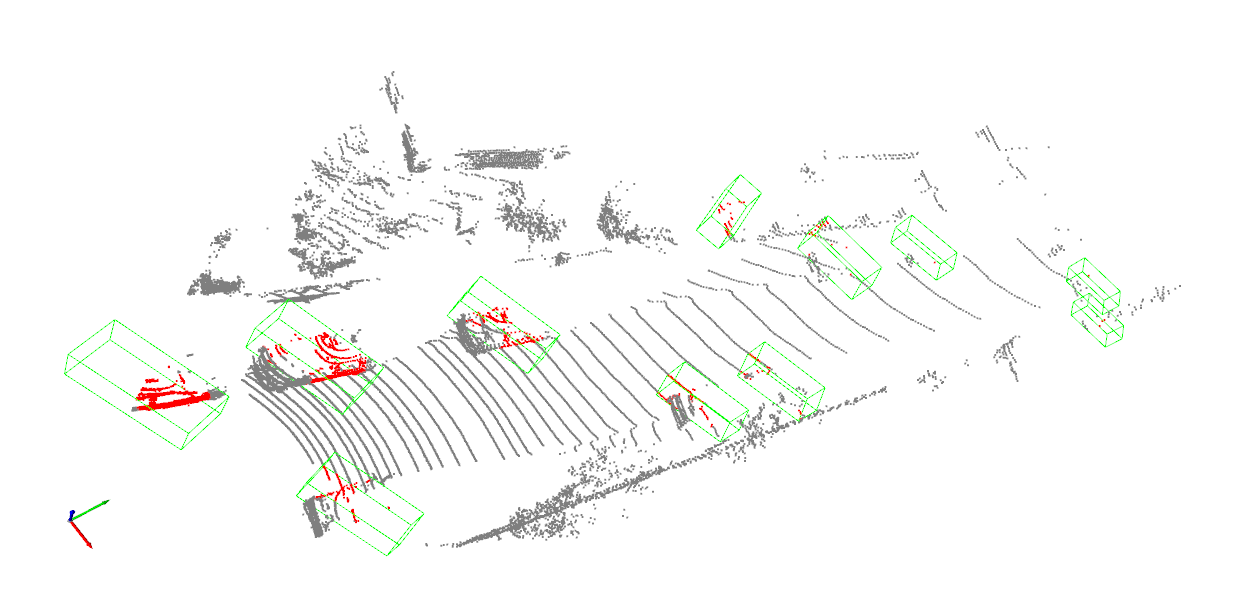


图6.2 输出结果

由图6.1可知，输入点云数据中共有13辆车，HV-SECOND车辆检测模型共检测出11辆车，如图6.2所示，未被检测出的2辆车都是位于街道的尽头，离数据采集点较远，且在图像上有大面积重叠的情况，在近距离即使出现重叠的情况，HV-SECOND车辆检测模型也能很好的检测出车的数量与位置。由此可知HV-SECOND车辆识别模型在复杂的较长街道的场景也能很好的识别出车辆，在该种情况下精度能够达到84.61%，近距离识别的精度能达到100%. 

图6.3 KITTI数据的二维场景图片

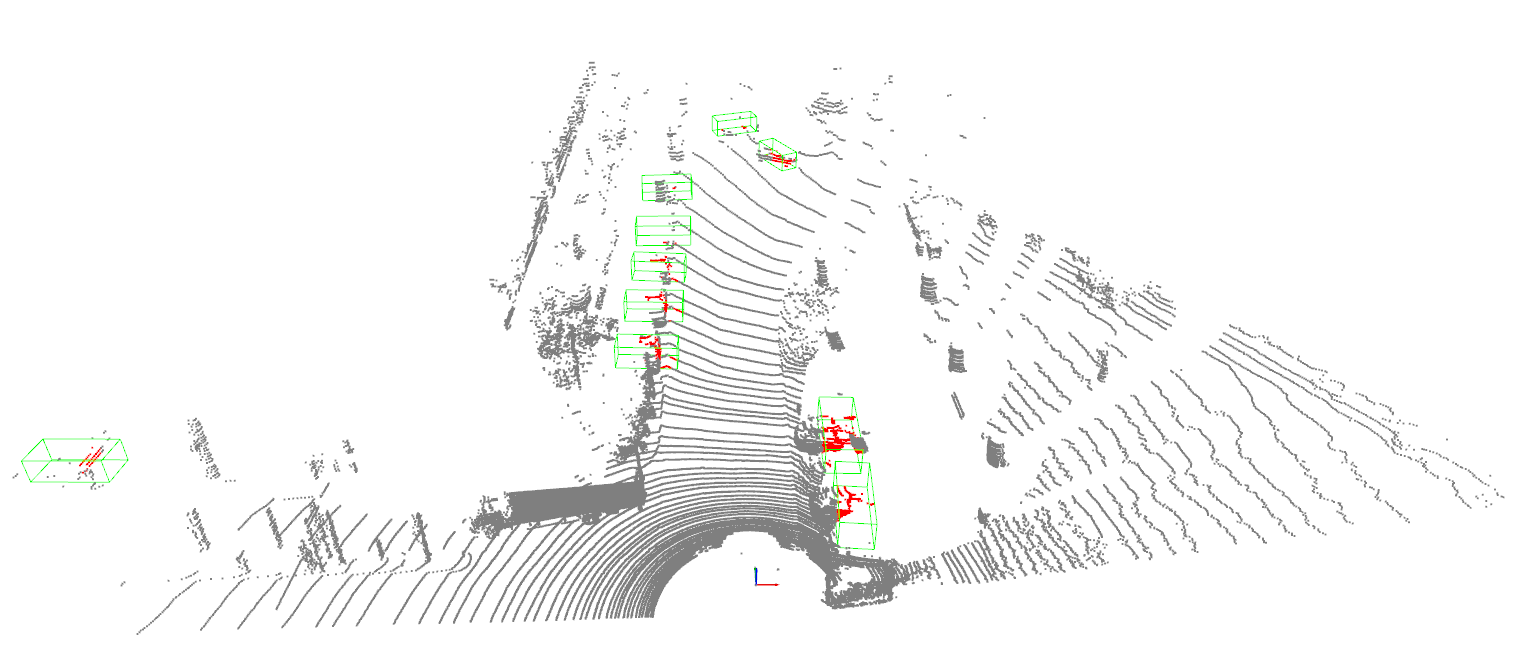


图6.4 输出结果

如图6.3所示，在小区停车位场景情况下，位于传感器较近的7辆汽车基本上都能够准确检测出来，如图6.4所示，距离较远的汽车则有一定的概率检测不出来。



图6.5 sunrgbd数据的图片形式

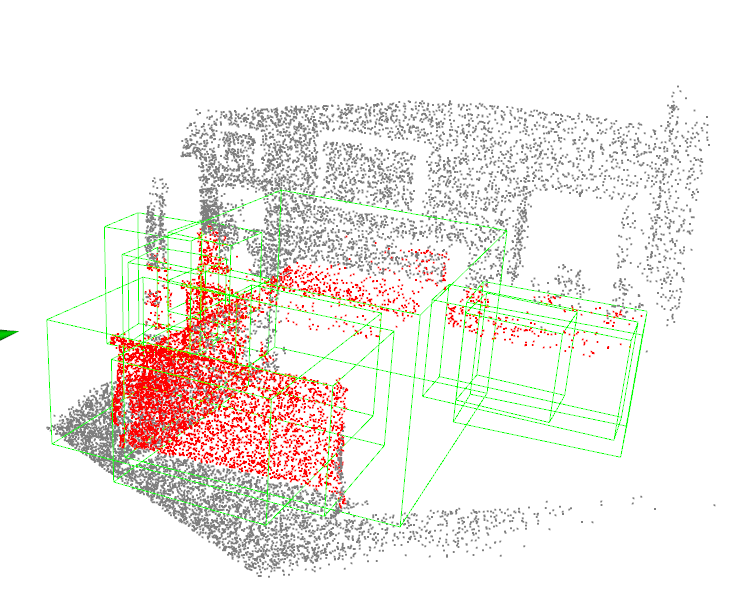


图6.6 VoteNet模型输出

由图6.6可知，软件对加载的模型和数据集得到了比较好的输出结果，输入点云数据中，床，床头的柜子，床头墙上的画，镜子都有准确的识别出来。

## 6.2 测试的结果分析

软件基于mmdetection3D的框架，能够很好的接受不同的模型，以及数据集，并且具备快速的对模型进行评估的一个能力，根据测试模型的不同，测试的时间也有所差异，见表6.1所示，软件改变了mmdetection3D框架加载参数的方式，让参数的加载，模型的加载以及数据集的加载变得更加方便，这极大的提高了算法评估者的工作效率。

表6.1 不同模型测试所需的平均时间

|  |  |  |
| --- | --- | --- |
| 模型种类 | 加载数量 | 平均时间 |
| HV-SECOND | 10 | 96秒 |
| VoteNet | 10 | 141秒 |

## 6.3 本章小结

本章完成了软件的测试环节，软件能够很好的容纳各种各样的模型和数据集，并对模型进行了准确的评估，测试出了模型的准确度，以及其优缺点，有的模型对于距离较近的目标有着较高的准确度，但是较远的精度较差，有的模型在环境比较复杂的室内环境中，仍然能够具备较高的检测精度，对不同类型的物品仍具备检测与分类的能力。并且使用该软件能够较为迅速的对不同的模型进行快速的评估，这样提高了算法评估者的工作效率。

# 结 论

本课题完成了一种基于三维点云数据的目标检测算法软件系统的设计与实现。该系统通过对三维点云数据进行分析和处理，能够实现对目标的检测和识别，并提供模型的更换与数据集加载的功能，并制作了GUI界面方便算法评估者的使用。

首先，文章介绍了三维点云数据的基本概念和处理方法，包括点云数据表示、滤波、配准和特征提取等，以及三维目标检测技术的主要方法和流程。接着，文章详细说明了相关目标检测系统的设计和实现过程，包括系统架构、模块功能和实现方法等，其中特别强调了算法优化和性能提升的关键技术。最后，文章对该系统进行了实验验证和性能评估，结果表明该系统具有较高的检测精度和鲁棒性，并且能够满足实时处理和多任务处理的需求。

本课题实现了该课题的要求，但是碍于自身水平的技术储备有限，没有设计一个更美观的界面，以及更多功能的集成，虽然能够完成基本的模型评估，但是没有提供模型下载的接口和数据集下载的接口，仍然给算法评估者的使用带来了一些不便。

综上，该文章介绍了一种基于三维点云数据的目标检测算法软件系统的设计与实现，是三维点云数据处理和目标检测领域的重要研究成果，具有广泛的应用前景和市场潜力。

# 

# [参考文献](file:///C:\\Users\\111\\Desktop\\论文格式摸板(2007).doc)

[1]单波.三维点云数据中的目标识别方法研究[D].深圳大学,2016.

[2]Shen B, Yin F, Chou W. A 3D Modeling Method of Indoor Objects Using Kinect Sensor[C]//2017 10th International Symposium on Computational Intelligence and Design (ISCID). IEEE, 2017,1: 64-68.

[3]Zheng J,Yang S,Wang X,et al. A decision tree based road recognition approach using roadside fixed 3D LiDAR sensors[J]. IEEE Access, 2019, 7: 53878-53890.

[4]金志伟,黄昶,祝瑞红.基于高重叠度视角的锁销点云模型的快速建立[J].华东师范大学学报 (自然科学版),2023(02):95-105.

[5]赵佳琦,周勇,何欣,卜一凡,姚睿,郭睿.基于深度学习的点云分割研究进展分析[J].电子与信 息学报,2022,44(12):4426-4440

[6]李娇娇,孙红岩,董雨,张若晗,孙晓鹏.基于深度学习的3D点云处理综述[J/OL].计算机研究 与展:1-24[2023-05-30].http://kns.cnki.net/kcms/detail/11.1777.TP.20210825.1539.005.html

[7]刘大峰,廖文和,戴宁,程筱胜.散乱点云去噪算法的研究与实现[J].东南大学学报(自然科学 版),2007(06):1108-1112.

[8]盛丽.基于3D激光点云的目标识别方法研究[D].哈尔滨工程大 学,2021.DOI:10.27060/d.cnki.ghbcu.2021.000772.

[9]钟志鹏,张建州,梁彪.基于OPTICS聚类和改进双边滤波的点云去噪方法[J].现代计算 机,2020(10):76-80.

[10]Jones T R, Durand F, Desbrun M. Non-iterative, feature-preserving mesh smoothing[M]//ACM SIGGRAPH 2003 Papers. 2003: 943-949.

[11]赵灿,董刚,程俊廷.利用噪声特点与曲面拟合的点云去噪及光顺算法研究[J].现代制造工 程,2008(06):90-93.

[12]张毅,刘旭敏,隋颖,关永.基于K-近邻点云去噪算法的研究与改进[J].计算机应 用,2009,29(04):1011-1014.

[13]葛宝臻,项晨,田庆国,彭博.基于曲率特征混合分类的高密度点云去噪方法[J].纳米技术与 精密工程,2012,10(01):64-67.

[14]曹爽,岳建平,马文.基于特征选择的双边滤波点云去噪算法[J].东南大学学报(自然科学 版),2013,43(S2):351-354.

[15]刘强.双振镜系统激光点云数据预处理[D].电子科技大学,2014.

[16]王增涛. 三维点云数据处理平台设计[D].大连理工大学,2014.

[17]郭毅锋,吴帝浩,魏青民.基于深度学习的点云三维目标检测方法综述[J].计算机应用研 究,2023,40(1):20-27

[18]张建民,陈富健,龙佳乐.基于图像处理的点云滤波算法[J].激光与光电子学进 展,2021,58(6):221-232

[19]柏东芳. 基于voxel表征的三维点云数据配准与目标检测[D].河北工业大学,2016.

[20]李建微,占家旺.三维点云配准方法研究进展[J].中国图象图形学报,2022,27(2):349-367

[21]马建红,王稀瑶,陈永霞,林楠.自动驾驶中图像与点云融合方法研究综述[J].郑州大学学报 (理学版),2022,54(06):24-33.DOI:10.13705/j.issn.1671-6841.2022102.

[22]康自祥,王升哲,崔雨勇,高欣仪,陈旺成.基于Transformer的体素化激光点云目标检测算法 [J].激光与红外,2023,53(2):202-207

[23]冯宇健. 基于深度度量学习的素描草图人脸识别[D].南京邮电大 学,2020.DOI:10.27251/d.cnki.gnjdc.2020.001498.

[24]王子明.面向点云语义分割的多特征融合PointNet++网络[J].地理空间信 息,2023,21(05):25-28.

[25]李超,陈秋帆.基于卷积神经网络的建筑空间改造智能辅助设计研究[J].自动化与仪器仪 表,2023(04):191-195.DOI:10.14016/j.cnki.1001-9227.2023.04.191.

[26]刘训华,孙韶媛,顾立鹏,李想.基于改进Frustum PointNet的3D目标检测[J].激光与光电子 学进展,2020,57(20):320-326

[27]宋一凡,张鹏,宗立波,马波,刘立波.改进的基于冗余点过滤的3D目标检测方法[J].计算机 应用,2020,40(9):2555-2560

[28]郝非凡,马翔越,李昊洋,刘忠富.自动驾驶汽车探测传感器及其融合技术综述[J].山西电子 技术,2022(03):93-96.

[29]王心怡. 基于特征学习的数字图像篡改盲取证技术研究[D].北京邮电大 学,2021.DOI:10.26969/d.cnki.gbydu.2021.000105.

[30]Lang,,Alex,H.,Vora,,Sourabh,Caesar,,Holger,Zhou,,Lubing,Yang,,Jiong,Beijbom,,& Oscar.(2018).PointPillars: Fast encoders for object detection from point clouds.arXiv.

# 致 谢

在此，我要向所有支持和帮助过我的人们表示最真挚的感谢。在我完成毕业论文的漫长路程中，无论是精神上还是实际上，有你们的大力支持，才让我能够度过这段艰辛的旅程。

首先，我要感谢我的导师，他在我的毕业论文研究中给予我了深入的指导和无私的帮助。他的启示和建议让我受益良多，重新审视并完善了我的研究方向和方法。

其次，我还要感谢我的家人，是他们一直以来的支持和关爱，鼓舞了我在遇到困难和挫折时坚定前行。他们是我坚实的后盾和支持，他们的爱激发了我不断进步的动力，是我最大的鼓励和支持。

然后，我还要感谢我的同学和朋友们，是他们一直以来的陪伴和支持鼓励，让我在失败和挫折中坚定自己的目标和信念。与他们长期的交流和讨论，让我不断提高自己的认识和见解，让我的论文更加深入、丰富和具有价值。

最后，我要感谢评委会老师的精心审阅和指导，他们的专业知识和严谨认真的态度给予了我很多宝贵的建议和意见，帮助我进一步提高自己的论文质量和提高自我价值。

感谢所有关心和支持我的人们。正是因为您们的支持和鼓励，我才能顺利完成我的毕业论文。谢谢大家！

作者简介：

姓 名：任宏培 性别：男

出生年月：2001年5月 民族：汉族

E.mail: 646245700@qq.com

# 

# 声 明

本论文的工作是2022年12月至2023年6月在成都信息工程大学自动化学院完成的。文中除了特别加以标注的地方外，不包含他人已经发表或撰写过的研究成果，也不包含为获得成都信息工程大学或其他教学机构的学位或证书而使用过的材料。

关于学位论文使用权和研究成果知识产权的说明：

本人完全了解成都信息工程大学有关保管使用学位论文的规定，其中包括：

（1）学校有权保管并向有关部门递交学位论文的原件与复印件。

（2）学校可以采用影印、缩印或其他复制方式保存学位论文。

（3）学校可以学术交流为目的复制、赠送和交换学位论文。

（4）学校可允许学位论文被查阅或借阅。

（5）学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

除非另有科研合同和其他法律文书的制约，本论文的科研成果属于成都信息工程大学。

特此声明！

作者签名：任宏培

2023年 5 月30 日

# 

# 附 录

完整功能程序如下：

软件登录界面：

from PyQt5 import QtCore, QtGui, QtWidgets, Qt

from PyQt5.QtGui import QIcon

from uitest import \*

from PyQt5.QtWidgets import \*

from PyQt5.QtCore import \*

class Ui\_MainWindow(QtWidgets.QMainWindow):

def \_\_init\_\_(self):

super(Ui\_MainWindow,self).\_\_init\_\_()

self.setupUi(self)

self.retranslateUi(self)

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(386, 127)

MainWindow.setWindowIcon(QIcon('logo.png'))

MainWindow.setStyleSheet("background-image:url(Background.jpg)")

self.centralWidget = QtWidgets.QWidget(MainWindow)

self.centralWidget.setObjectName("centralWidget")

self.lineEdit = QtWidgets.QLineEdit(self.centralWidget)

self.lineEdit.setGeometry(QtCore.QRect(250, 20, 100, 20))

self.lineEdit.setText("")

self.lineEdit.setObjectName("lineEdit")

self.lineEdit\_2 = QtWidgets.QLineEdit(self.centralWidget)

self.lineEdit\_2.setGeometry(QtCore.QRect(250, 50, 100, 20))

self.lineEdit\_2.setText("")

self.lineEdit\_2.setEchoMode(QtWidgets.QLineEdit.Password)

self.lineEdit\_2.setObjectName("lineEdit\_2")

self.label = QtWidgets.QLabel(self.centralWidget)

self.label.setGeometry(QtCore.QRect(200, 24, 32, 18))

self.label.setTextFormat(QtCore.Qt.AutoText)

self.label.setObjectName("label")

self.label\_2 = QtWidgets.QLabel(self.centralWidget)

self.label\_2.setGeometry(QtCore.QRect(200, 54, 32, 18))

self.label\_2.setObjectName("label\_2")

self.pushButton = QtWidgets.QPushButton(self.centralWidget)

self.pushButton.setGeometry(QtCore.QRect(190, 90, 75, 23))

self.pushButton.setObjectName("pushButton")

self.pushButton\_2 = QtWidgets.QPushButton(self.centralWidget)

self.pushButton\_2.setGeometry(QtCore.QRect(290, 90, 75, 23))

self.pushButton\_2.setObjectName("pushButton\_2")

MainWindow.setCentralWidget(self.centralWidget)

self.pushButton.clicked.connect(self.word\_get)

self.pushButton\_2.clicked.connect(MainWindow.close)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "登录系统"))

self.lineEdit.setPlaceholderText(\_translate("MainWindow", "请输入帐号"))

self.lineEdit\_2.setPlaceholderText(\_translate("MainWindow", "请输入密码"))

self.label.setText(\_translate("MainWindow", "帐号"))

self.label\_2.setText(\_translate("MainWindow", "密码"))

self.pushButton.setText(\_translate("MainWindow", "确定"))

self.pushButton\_2.setText(\_translate("MainWindow", "取消"))

def word\_get(self):

login\_user = self.lineEdit.text()

login\_password = self.lineEdit\_2.text()

if login\_user == 'admin' and login\_password == '123456':

ui\_hello.show()

MainWindow.close()

else:

QMessageBox.warning(self,

"警告",

"用户名或密码错误！",

QMessageBox.Yes)

self.lineEdit.setFocus()

if \_\_name\_\_ == "\_\_main\_\_":

import sys

app = QtWidgets.QApplication(sys.argv)

MainWindow = QtWidgets.QMainWindow()

ui = Ui\_MainWindow()

ui\_hello = hello\_mainWindow()

ui.setupUi(MainWindow)

MainWindow.show()

sys.exit(app.exec\_())

# 创新项目

# 编程：任宏培

# 开发时间：2023/4/11 11:31

# 创新项目

# 编程：任宏培

# 开发时间：2023/3/22 17:17

import sys

from PyQt5.QtWidgets import QApplication, QMainWindow, QAction, QFileDialog, QPushButton, QMessageBox

from argparse import ArgumentParser

import open3d as o3d

import numpy as np

from mmdet3d.apis import inference\_detector, init\_model, show\_result\_meshlab

from PyQt5.QtWidgets import QApplication, QMainWindow

class hello\_mainWindow(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

# 设置窗口标题

self.setWindowTitle("基于三维点云的目标检测算法的软件系统设计与实现")

self.resize(500, 500)

# 创建菜单栏

menu\_bar = self.menuBar()

# 创建文件菜单

file\_menu = menu\_bar.addMenu("文件")

# 创建打开数据集动作

open\_dataset\_action = QAction("打开数据集", self)

open\_dataset\_action.triggered.connect(self.open\_dataset)

# 创建打开模型动作

open\_model\_action = QAction("打开模型", self)

open\_model\_action.triggered.connect(self.open\_model)

# 创建打开py动作

open\_py\_action = QAction("打开py文件", self)

open\_py\_action.triggered.connect(self.open\_py)

# 创建打开模型动作

change\_data\_action = QAction("转换数据类型", self)

change\_data\_action.triggered.connect(self.change\_data)

# 创建退出动作

exit\_action = QAction("退出", self)

exit\_action.triggered.connect(self.close)

# 添加动作到文件菜单

file\_menu.addAction(open\_dataset\_action)

file\_menu.addAction(open\_model\_action)

file\_menu.addAction(open\_py\_action)

file\_menu.addAction(change\_data\_action)

file\_menu.addAction(exit\_action)

# 创建按钮

self.start\_button = QPushButton("开始执行", self)

self.start\_button.setGeometry(125, 225, 200, 50)

self.start\_button.clicked.connect(self.start\_execution)

# 初始化数据集、模型和py文件的路径

self.dataset\_path = None

self.model\_path = None

self.py\_path = None

self.data\_path = None

def open\_dataset(self):

# 打开数据集对话框

file\_dialog = QFileDialog()

file\_dialog.setFileMode(QFileDialog.AnyFile)

file\_dialog.setNameFilter("bin文件 (\*.\*)")

if file\_dialog.exec\_():

filenames = file\_dialog.selectedFiles()

self.dataset\_path = filenames[0]

print("打开数据集:", self.dataset\_path)

def open\_model(self):

# 打开模型对话框

file\_dialog = QFileDialog()

file\_dialog.setFileMode(QFileDialog.AnyFile)

file\_dialog.setNameFilter("模型文件 (\*.\*)")

if file\_dialog.exec\_():

filenames = file\_dialog.selectedFiles()

self.model\_path = filenames[0]

print("打开模型:", self.model\_path)

def open\_py(self):

# 打开模型对话框

file\_dialog = QFileDialog()

file\_dialog.setFileMode(QFileDialog.AnyFile)

file\_dialog.setNameFilter("py文件 (\*.\*)")

if file\_dialog.exec\_():

filenames = file\_dialog.selectedFiles()

self.py\_path = filenames[0]

print("打开py:", self.py\_path)

def change\_data(self):

# 打开模型对话框

file\_dialog = QFileDialog()

file\_dialog.setFileMode(QFileDialog.AnyFile)

file\_dialog.setNameFilter("txt文件 (\*.\*)")

if file\_dialog.exec\_():

filenames = file\_dialog.selectedFiles()

self.py\_path = filenames[0]

print("打开txt:", self.data\_path)

txt\_path = self.data\_path

# 通过numpy读取txt点云

pcd = np.genfromtxt(txt\_path, delimiter=",")

pcd\_vector = o3d.geometry.PointCloud()

# 加载点坐标

pcd\_vector.points = o3d.utility.Vector3dVector(pcd[:, :3])

o3d.visualization.draw\_geometries([pcd\_vector])

point = np.asanyarray(pcd\_vector.points)

size\_point = len(point)

point = np.column\_stack((point, np.arange(1, size\_point + 1)))

point[:, 3] = 0

point.astype(np.float32).tofile('test.bin')

np.save("test.npy", point)

def start\_execution(self):

if self.dataset\_path is None:

QMessageBox.warning(self, "警告", "请先选择数据集")

return

if self.model\_path is None:

QMessageBox.warning(self, "警告", "请先选择模型")

return

if self.py\_path is None:

QMessageBox.warning(self, "警告", "请先选择py文件")

return

# 执行代码

print("开始执行")

print('数据集', self.dataset\_path)

# build the model from a config file and a checkpoint file

model = init\_model(self.py\_path, self.model\_path, device='cuda:0')

# test a single image

result, data = inference\_detector(model, self.dataset\_path)

# show the results

show\_result\_meshlab(

data,

result,

"output",

0.0,

True,

False,

task='det')

if \_\_name\_\_ == '\_\_main\_\_':

app = QApplication(sys.argv)

window = hello\_mainWindow()

window.show()

sys.exit(app.exec\_())