分类号：　　密级：
U D C：　　学号：416100210156

南 昌 大 学 专 业 学 位 硕 士 研 究 生

学 位 论 文

基于激光雷达与相机融合的室外目标实时检测与跟踪方法研究

Research on Real-time Detection and Tracking Method of Outdoor Targets Based on LiDAR and Camera Fusion

熊志君

培养单位（院、系）：信息工程学院　电子信息工程系

指导教师姓名、职称：陈利民　副教授

指导教师姓名、职称：罗齐熙　高级工程师

专业学位种类：电子信息

专业领域名称：

论文答辩日期：2024 年 6 月 4 日

答辩委员会主席：_____

评阅人：_____

_____

2024 年 6 月 4 日

摘　要

目标检测与跟踪是自主导航机器人环境感知的关键技术之一，是机器人精准态势感知、预测、决策的基础。室外环境复杂多变，使用单一传感器获取信息时容易受到自身限制。为此，结合深度神经网络设计了一种新的激光雷达与相机融合的目标检测与跟踪方法，具体工作内容如下所列。

针对现有基于深度神经网络的视觉目标检测算法高复杂度与嵌入式平台低算力的矛盾，提出一种轻量化的目标检测算法。首先，结合 GhosNet 网络减小模型参数规模。然后，在骨干网络中融入无参注意力机制在不增加参数规模的情况下增强模型特征提取能力。最后，引入 SimSPPF 激活函数提高模型实时性。在 KITTI 数据集上进行测试，mAP 提升 2.35%，且模型参数量下降 46.6%。

针对点云小目标漏检、误检的问题，提出一种改进的 PointPillars 激光点云目标检测算法。首先，引入高效通道注意力机制优化二维卷积特征提取网络的设计，实现伪图像特征中位置特征信息的增强和背景噪声等不相关特征信息的弱化。然后，引入 Softplus 激活函数来提高模型对负值信息的处理能力。最后，通过优化目标朝向损失，进一步提升模型的精度和鲁棒性。在 KITTI 数据集上测试所提出算法的性能，该算法在 mAP 上提升 4.57%。

针对轻量化平台算力低的特点，构建一种低复杂度的决策级融合算法。该算法通过融合两种传感器的信息，弥补了单一传感器的局限性，提高了目标检测精度。检测到目标后，利用 DeepSORT 算法进行目标跟踪。经过 KITTI 公共数据集上的测试验证，相较于单一传感器的方法，所提方法在遮挡、光线不佳等复杂环境下能更好的完成目标检测与跟踪任务。

关键词：激光点云；注意力机制；轻量化深度神经网络；信息融合识别；目标跟踪

ABSTRACT

Object detection and tracking are crucial technologies for environment perception in autonomous navigation robots, forming the basis for precise situational awareness, prediction, and decision-making. The outdoor environment is complex and constantly changing, making it challenging to rely on a single sensor for information acquisition due to inherent limitations. To address this, a novel method combining a depth neural network with a fusion of LiDAR and camera data has been designed for object detection and tracking. The specific tasks involved are listed below.

To address the contradiction between the high complexity of existing deep neural network-based object detection algorithms and the low computational power of embedded platforms, a lightweight object detection algorithm is proposed. Firstly, the model parameter size is reduced by combining the GhosNet network. Next, a parameter-free attention mechanism is integrated into the backbone network to enhance the model's feature extraction capabilities without increasing the parameter size. Finally,the SimSPPF activation function is introduced to improve real-time performance of the model. Testing on the KITTI dataset shows a 2.35% improvement in mAP, with a reduction of 46.6% in model parameter count.

In response to the issues of missed detection and false alarms in small object detection, an improved PointPillars laser point cloud object detection algorithm is proposed. Firstly, an efficient channel attention mechanism is introduced to optimize the design of the 2D convolution feature extraction network, enhancing position feature information in pseudo-image features while weakening irrelevant features such as background noise. Next, the Softplus activation function is introduced to improve the model's handling of negative information. Finally, by optimizing the target orientation loss, the model's accuracy and robustness are further improved. Testing on the KITTI dataset shows a 4.57% improvement in mAP for the proposed algorithm.

In response to the low computing power of lightweight platforms, a low-complexity decision-level fusion algorithm is constructed. This algorithm combines

information from two sensors to compensate for the limitations of a single sensor, thereby improving target detection accuracy. Upon detecting a target, the DeepSORT algorithm is employed for target tracking. Through testing and validation on the KITTI public dataset, the proposed method shows better performance in target detection and tracking tasks in complex environments such as occlusions and poor lighting conditions compared to single-sensor methods.

Key words: Laser point cloud; Attention mechanism; Lightweight deep neural network; Information fusion recognition; Target tracking

目 录

第1章 绪论

1.1 课题的研究背景及意义

随着时代的发展，机器人技术正在以前所未有的速度向前推进，并深入到社会生活的各个角落。无论是制造业、采矿业、建筑业等传统行业，还是家庭服务、医疗健康、养老助残等新兴领域，机器人都发挥着越来越重要的作用。

图 1.1 不同用途机器人

环境感知为自主导航机器人提供定位、避障及路径规划所需的环境信息，其中目标检测与跟踪技术作为感知系统的重要组成成分，一直是自主导航机器人领域的重要研究热点[1]。这些技术赋予了自主导航机器人对周围环境的深入感知能力，使其能够获取目标的运动轨迹、速度和方向等关键信息[2]。通过这些信息，自主导航机器人能够更准确地理解环境动态，从而做出更明智的决策和行动。然而，目标检测与跟踪技术在实际应用中也面临着诸多挑战。

相机是自主导航机器人搭载的一种常见的传感器，能够提供关于环境中物体的足够语义信息。但是，相机也有一定的局限性，其缺乏对检测到的物体的详细测量信息。在检测到物体之后，很难精确定位和测量物体的尺寸和方向。此外，相机对光线和镜面反射很敏感，这影响其在夜间的正常功能[3]。激光雷达能够获得准确的 3D 空间信息，且不受外界光线等因素的干扰，但是对于物体类别的判断效果较差。除此之外，雨雪等极端天气情况也会对激光雷达的效果产生一定的影响，使得其性能有所下降。

基于上述问题，目前主流的方法是信息融合，即通过整合多模态传感器的数据，充分利用各传感器的互补优势[4]。这一方法运用统计模型或机器学习技术，将来自多个传感器的信息融合，生成对周围环境中对象更准确和全面的表示。在自主导航机器人的技术发展中，实现复杂环境下目标检测与跟踪具有至关重要的意义。为实现这一目标，多传感器融合技术成为了一条不可或缺的发展路径。

1.2 国内外研究现状

1.2.1 目标检测研究现状

（1）图像目标检测

随着时代的高速发展，基于图像的目标检测被应用于各个领域，包括火车站台、政府组织、工业生产。目标检测算法主要可分为两大类别：传统目标检测算法和基于深度学习的目标检测算法。传统算法在特征明显、背景简单的场景下表现良好。然而，在背景复杂多变，目标形态各异的情况下，传统方法难以仅凭抽象特征准确检测目标。相比之下，深度学习算法能够提取目标的丰富特征，能够更好地应对复杂多变的目标检测任务，因此在实际应用中更具优势。

深度学习模型通常包括编码器和解码器两个关键部分。编码器负责接收图像输入并通过深度学习模型的隐藏层提取统计特征，这些特征可用于图像的检测和标记。解码器则从编码器获取这些特征，并应用边界框等技术对图像进行标记[5]。对象检测的过程包括将视觉输入划分成多个区域，对每个区域进行单独处理，通过卷积神经网络获取可能的标签和标记。最后，将所有处理过的区域组合成一个图像，形成最终的输出。基于深度学习的目标检测任务通常被分为两大类：两阶段检测器和一阶段检测器。二阶段检测器顾名思义分成两个阶段。在第一阶段，两阶段网络主要致力于生成候选区域；第二阶段将这些候选区域送入更精细的深度学习模型中进行预测和分类[6]。二阶段目标检测算法有 R-CNN[7]、SPP-Net[8]、Fast R-CNN[9]、Faster R-CNN[10]、Mask R-CNN[11]等。这些算法可以在不需要任何提示的情况下轻松获得高精度检测结果，但由于速度慢和结构复杂，在工程中应用的较少。由于一阶段检测器在检测过程中没有进行候选区域的筛选和精细调整，其检测精度相对于两阶段检测器略有不足。为了弥补这一不足，研究者们后续提出了一些精度更高的算法，用于改进一阶段检测器的性能。2015 年

Redmon 等人[12]提出 YOLOv1 算法，将对象检测视为空间分离的边界框和相关类概率的回归问题。只需要一个网络就能完成定位和分类两个任务，在检测精度和速度上也有着显著的优势。同年 Erhan 等人[13]提出了 SSD，SSD 是从 YOLOv1 基础改进而来，不同的是 SSD 提取不同尺度的特征图来做检测。此外。还采用不同尺度和长宽比的先验框。RetinaNet[14]在 2017 年被提出，是一个里程碑性质的模型。在目标检测领域，作为一个一阶段网络，首次超过了当时流行的二阶段网络模型。RetinaNet 通过引入 Focal Loss 重塑标准交叉熵损失，动态调整不同样本的损失权重，使得模型更加关注难以分类的样本，从而有效解决类别不平衡问题。Adarsh 等人[15]提出 YOLOv3-Tiny，该算法采用了更深的网络结构和多尺度检测策略，可以检测不同尺度的目标。在 2020 年，YOLOv5 发布，由于采用了轻量级网络结构和新的训练策略，可以在更快的速度下实现高精度目标检测。最近 YOLO 系列算法发展势头强劲，一直不断优化与创新，得到了各界人士广泛的应用和认可[16-18]。

（2）点云目标检测

近年来，随着三维技术的深入探索与持续进步，依赖三维点云数据的目标识别技术也逐渐崭露头角，受到了越来越多人的关注。点云数据包含了物体表面的三维坐标信息，因此能够准确地描述物体的形状、尺寸和位置等空间特性。这使得点云目标检测能够更精确地定位目标物体，大幅提升了物体检测的准确性。目前，广受关注的 3D 检测方案主要使用激光雷达传感器进行目标检测。激光雷达通过发射和接收激光脉冲，在其扫描范围内获取 3D 点云数据，从而描述和呈现三维世界的结构和特征[19]。激光因其出色的方向性和能量集中特性，使得 3D 点云数据极为精确，同时拥有极高的空间分辨率。激光雷达作为一种有源成像系统，同无源光学和有源雷达或微波仪器相比，不受光照条件的限制，故而在复杂环境下三维物体检测方面具有天然优势。近年来，基于激光雷达的三维点云目标检测吸引了越来越多的关注。这类算法主要分为传统算法和基于深度学习算法两大类别。

传统的点云目标检测一般包括数据预处理、点云分割，障碍物聚类和目标分类等过程[20]。在目标检测中，目标分类是一个至关重要的环节。然而，传统算法在处理多目标分类任务时存在一定的局限性。这些算法通常需要预先定义目标的类别，这在面对多样化或未知目标时，显得有些不足。此外，在复杂场景中，自然图像特征提取的准确率也是一个技术难题。幸运的是，近年来深度学习和传

感器技术的突破性进展为机器人感知技术带来了全新的发展机遇。然而，即便有这些技术的支持，要实现对三维环境的准确理解和实时感知仍然面临着一系列挑战。这包括对数据进行高质量处理的需求，以消除噪声和干扰；在复杂场景下准确识别目标的能力，以应对各种突发情况；以及高效决策制定的需求，以确保感知系统能够在短时间内做出正确的反应。目前主流的基于点云的三维检测方法主要是 point-based 和 voxel-based 两种。在 21 世纪初之前，对点云研究一般都是将此类数据转换为规则的 3D 体素网格或图像集合。2017 年 Charles 等人[21]提出 PointNet，能够直接对无序的点云数据进行处理，并提取出有意义的特征。PointRCN[22]在 PointNet 工作上推出了一个新的点云三维目标检测框架，该框架由两个阶段组成：stage-1 用于自下而上的 3D 提案生成，stage-2 用于在规范坐标中细化提案以获得最终的检测结果。原作者又从 PointRCNN 扩展得到 Part-A2[23]，整个框架使用部分感知和部分聚合两个模块实现 3D 目标检测任务。2020 年 Yang 等人[24]提出 3DSSD，3DSSD 不同于以往的 point-based 算法，其去掉了 FP 层和 RM(Refinement Module)，以便于加快运行速度。除此之外，在融合采样方法和预测框网络上做出了创新。voxel-based 采用一种与 point-based 不同的方法，选择将点云划分成一个个的网格，使用网格作为一个点云单位。VoxelNet[25]将点云划分为等间距的 3D 体素，并通过新引入的体素特征编码(Voxel Feature Encoding, VFE)将每个体素内的一组点转换为统一的特征表示。通过这种方式，点云被编码为描述性体积表示，然后连接到区域生成网络(Region Proposal Network,Region Proposal,PN)以生成检测。Second[26]通过使用稀疏卷积，大大减少了计算资源。PointPillars[27]进一步对计算资源进行优化，通过将点云划分成点柱，然后使用二维卷积处理 2D 伪图像。PointGNN[28]将点云数据转换为图结构，并开发了一种基于多顶点融合的合并算法来预测结果。SASSD[29]使用一种新方法解决 3D 卷积损失空间信息的问题，即利用辅助网络将卷积特征转化为点集的表现形式。在后续的研究中，有一些研究者将 Point-based 和 voxel-based 进行结合，创造了一些性能优越的检测算法。Fast PointRCNN[30]将轻量级卷积卷积后的网格化特征与点的特征采用注意力机制进行结合，这种方式保留了点云的空间信息同时也提供了准确的坐标位置。PVRCNN[31]采用多尺度提取的方法得到高精度的 voxel 特征，再利用 PointNet 获取更精细的点云局部信息，巧妙地结合了两种算法的优点。

1.2.2 传感器融合研究现状

多传感器融合是综合不同传感器信息以克服单一传感器局限性和不确定性，从而更全面、准确地感知与识别环境或目标，提升系统外部感知能力，确保在复杂环境中获取可靠、精确信息的过程。

早在 1998 年国外学者 Llinas[32]对多传感器融合在国防和非国防领域的重要性做了叙述。2002 年 Coué 等人[33]提出使用贝叶斯概率推理技术解决多传感器数据融合这一具有挑战性问题。2004 年 Sun 等人[34]提出一种多传感器最优信息融合准则，该准则针对具有多个传感器和相关噪声的离散时间线性随机控制系统，给出了一种具有两层融合结构的通用多传感器最优信息融合离散卡尔曼滤波器。2019 年 Zhou 等人[35]通过修改 YOLOv2 神经网络来实现毫米波雷达与摄像头之间信息的深度融合。2023 年 Senel 等人[36]针对环境感知问题提出使用决策级融合方法融合传感器的检测结果，然后使用匈牙利算法完成检测结果之间的匹配。

随着技术的快速迭代，多传感器信息融合效果日益优化，为机器人提供了更好的解决方案。这种技术融合不仅提高了机器人的安全性和稳定性，还为其在各种复杂环境中自主导航提供了强大的支持。

1.2.3 目标跟踪研究现状

目标跟踪就是给第一帧检测对象上加上一个 ID，然后在对象保持 ID 不变的基础上估计出对象在下一帧的位置和运动状态。主流的目标跟踪算法可分为基于激光点云的目标跟踪和基于图像的目标跟踪。

（1）基于图像的目标跟踪

使用相机进行目标跟踪的时候，常见的做法是先使用卡尔曼滤波器预测轨迹，然后使用 IOU(Intersection over Union)作为代价函数构建成本矩阵，最后运用匈牙利算法求解数据关联问题，从而实现精准的目标跟踪。2016 年 Bewley 等人[37]提出 SORT 算法，SORT 算法使用卡尔曼滤波器和匈牙利算法等熟悉技术，但该方法的精度可与最先进的在线跟踪媲美。2017 年 Wojke 等人[38]提出了 DeepSORT，这是对 SORT 算法的进一步改进。DeepSORT 算法通过集成外观信息来改善 SORT 的性能，使得即使在目标长时间遮挡后，仍然持续稳定地跟踪。

这一扩展不仅提高了跟踪的持久性，还有效减少了标识切换的次数，从而显著提升了目标跟踪的准确性和稳定性。

（2）基于激光点云的目标跟踪

激光雷达可以提供精准的三维空间信息，随着近几年深度学习技术的不断突破，基于点云的深度学习也聚焦了越来越多人的注意。Petrovskaya 等人[39-40]提出了一种基于运动特征和贝叶斯滤波器的跟踪算法，先建立目标的运动特征模型，再采用贝叶斯滤波器来对目标的运动状态进行估计和更新。Tanzmeister 等人[41-42]基于 DS 证据理论设计占用网格图，通过对各目标同时设置静态和动态占用概率模型，实现对动态、静态物体的跟踪。Held 等人[43]提出了一个实时 3D MOT 系统，系统将获得的激光雷达点云进行 3D 目标检测，然后使用 3D 卡尔曼滤波器和匈牙利算法的组合进行状态估计和数据关联。但是点云数据缺乏颜色等表面细节信息，导致在识别具有相似几何形状的不同类别对象时出现一定问题。例如，在道路上使用激光雷达进行目标跟踪很容易会将路边的花草或者路灯错误识别成行人，从而增加了误判的风险。

1.3 本文主要内容及章节安排

1.3.1 主要研究内容

传统单模态系统存在局限，使用单一传感器获取信息时容易受到自身限制。例如，相机能够提供关于环境中物体丰富的语义信息，然而在光照、遮挡等因素影响下成像质量会受到严重影响。激光雷达能够获得准确的 3D 空间信息，但对于物体类别的判断效果较差。针对这个问题，设计了一种新的激光雷达与相机融合的目标检测与跟踪方法，工作内容如下所列：

（1）提出一种轻量化的目标检测算法，该算法致力于解决基于深度神经网络的视觉目标检测算法高复杂度与嵌入式平台低算力的矛盾。首先，结合 GhosNet 网络减小模型参数规模。然后，在骨干网络中引入无参注意力机制(SimAM)在不增加参数规模的情况下增强模型特征提取能力。最后，使用 SimSPPF 激活函数提高模型实时性。

（2）提出了一种改进的 PointPillars 激光点云目标检测算法，该算法致力于解决远处点云过于稀疏导致算法无法识别的难题。首先，在二维卷积特征提取网

络中引入高效通道注意力机制(ECA-Net)，增强对伪图像特征中重要信息的提取能力。然后，引入 Softplus 激活函数来提高模型对信息的处理能力。最后，通过优化目标朝向损失，进一步提升了模型的精度和鲁棒性。同时，为了减轻嵌入式平台的计算负担，采用了体素滤波的方法对原始激光点云数据进行预处理。

（3）针对轻量化平台算力低的特点，构建一种低复杂度的决策级融合算法。该算法通过设计的决策级融合策略完成点云和视觉目标匹配，融合两种传感器的信息，弥补了单一传感器的局限之处，提高了检测精度。检测到目标后，利用 DeepSORT 算法进行目标跟踪。

1.3.2 论文结构安排

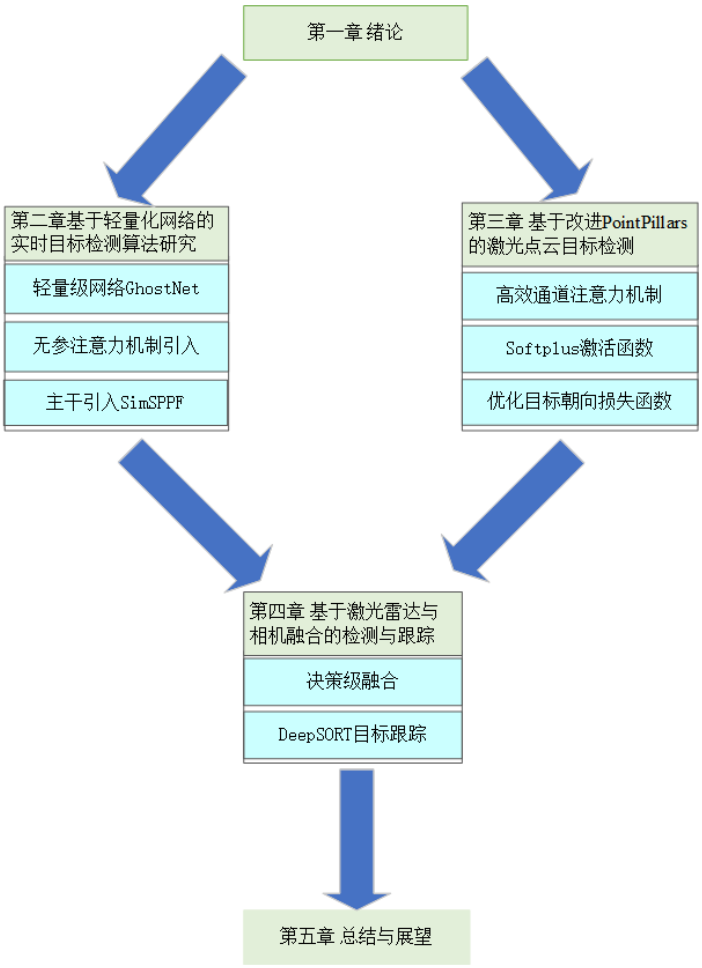


图 1.2 论文结构安排

第一章，绪论。首先介绍了论文的背景及意义，然后分别阐述了目标检测、多传感器融合与目标跟踪的国内外研究现状，最后介绍了本文的主要工作与章节安排。

第二章，基于轻量化网络的实时目标检测算法研究。首先，通过引入 GhostNet 轻量级模型减少模型的参数。然后，在骨干网络中引入 SimAM 注意力机制，在不增加参数数量的情况下增强模型特征提取能力。最后，引入 SimSPPF 激活函数，使模型能够更高效地进行特征提取，从而在保持准确性的同时提升了运行速度。

第三章，基于改进 PointPillars 的激光点云目标检测。首先，在二维卷积特征提取网络中添加了 ECA 注意力机制。ECA 模块的使用，可以使网络聚焦于伪图像特征中的重要信息，忽略无关信息。在提升模型的表达力的同时不会给嵌入式平台性能造成过多的额外负担。然后，为了捕获更多的有效信息，引入了 Softplus 激活函数，相较于 ReLU 激活函数，其具备更为丰富的特性。最后，对目标朝向损失函数进行优化，增强系统的稳定性和鲁棒性。

第四章，基于激光雷达与相机融合的检测与跟踪。设计了一种决策级融合方案，将激光雷达和相机识别结果进行决策级融合，得到更加完善的目标信息，确定目标后使用 DeepSORT 算法进行目标跟踪。

第五章，总结与展望。全文的主要工作与不足之处，介绍了进一步的研究方向与动态多传感器融合目标检测与跟踪的未来方向的展望。

第2章 基于轻量化网络的实时目标检测算法研究

2.1 引言

基于深度学习的视觉目标检测算法可以在复杂环境下出色完成特征提取任务，但经典的深度学习网络模型过于复杂，在嵌入式平台部署较为困难。基于此问题提出一种轻量化的目标检测算法，该算法致力于解决基于深度神经网络的目标检测算法高复杂度与嵌入式平台低算力的矛盾。首先，使用 Ghost 卷积和 C3Ghost 代替普通卷积和 C3 模块，以加快网络推理速度且减少参数和计算量。然后，在骨干网络中引入 SimAM 注意力机制在不增加参数规模的情况下增强模型特征提取能力。最后，结合 SimSPPF 激活函数进一步提高模型实时性。

2.2 基于深度学习的图像检测算法

视觉目标检测需要关注图片中特定目标的位置和类别。一个检测任务包括两个子任务：输出对象目标的类别信息和输出目标的具体位置信息。主流视觉检测算法分成两类，一类是传统方法，另一种是深度学习方法。传统需要手动设计特征提取器，在面对复杂非线性问题的时候，由于特征的设计无法充分表达出目标对象的特征，会导致检测准确性下降。基于深度学习的方法使用端到端的学习方式，直接从神经网络中学习特征，在处理非线性复杂问题的时候具有很强的适应性[44]。目前随着硬件资源的发展，基于深度学习的目标检测已经成为了主流检测算法。从网络的架构上区分，分成双阶段和单阶段检测器。双阶段物体检测器先通过提议生成阶段产生潜在目标的候选区域，然后再经过分类和定位阶段对这些候选区域进行进一步的处理，这样的体系保证了良好的精确度，但是也带来了检测效率低下的问题。一阶段网络直接从输入图像中预测目标的位置和类别，而无需生成候选区域，在处理速度上有着极大的优势。本节将分别以 R-CNN 系列和 YOLO 系列算法为例对两条路线作具体阐释说明。

2.2.1 双阶段目标检测算法

R-CNN 是二阶段网络中极具代表性的算法，在 2014 年被 Ross Girshick 提

出。相较于传统图像算法，R-CNN 使用基于候选区域的方法替代传统算法中滑动窗口[45]来寻找图像中可能存在目标的区域，使用卷积神经网络替代人工设计的特征用于目标特征的提取，然后使用支持向量机判断目标类别并使用边框回归修正候选框的位置。

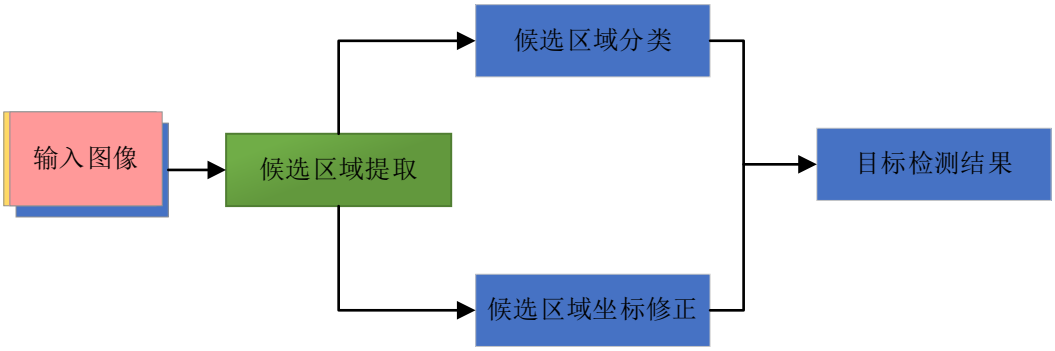


图 2.1 二阶段目标检测算法

RCNN 相对比传统算法检测精度提升了 30%左右，但是 RCNN 的检测流程包括多阶段处理，涉及候选区域生成、特征提取、分类和边界框微调等多个步骤，导致流程繁琐且难以优化。此外，RCNN 对每个候选区域进行独立特征提取，存在大量重复计算，且 CNN 本身计算复杂度高，导致整体计算力效率低下，限制了实时性和资源受限环境中的应用。

2.2.2 单阶段目标检测算法

在 2015 年，Redmon 等人提出了具有里程碑意义的 YOLO 算法，这一算法在目标检测领域引起了广泛关注。YOLO 算法摒弃了传统二阶段检测算法中候选框与分类器组合的复杂策略，开创性地将分类问题转化为回归问题，从而简化了目标检测的流程。

YOLO 算法通过构建一个端到端的卷积神经网络模型，实现了对图像中物体边界框和类别概率的直接预测。这一创新策略使得 YOLO 能够在单次前向传播过程中完成整个检测任务，无需像二阶段算法那样先生成候选区域再进行分类。这种一体化的设计不仅提高了检测速度，还有助于实现更加精确的目标定位。如图 2.2 所示，YOLO 在推断阶段经过网络直接将图像特征划分成 N×N 个待检测网格，N 的大小与图像的分辨率大小和研究人员对模型的配置情况相关，当一目标落入其中一个待检测网格中，该网格就执行识别该目标的任务。从每一个网

格生成多个候选锚框，通过模型预测判断目标物体是否在其中。

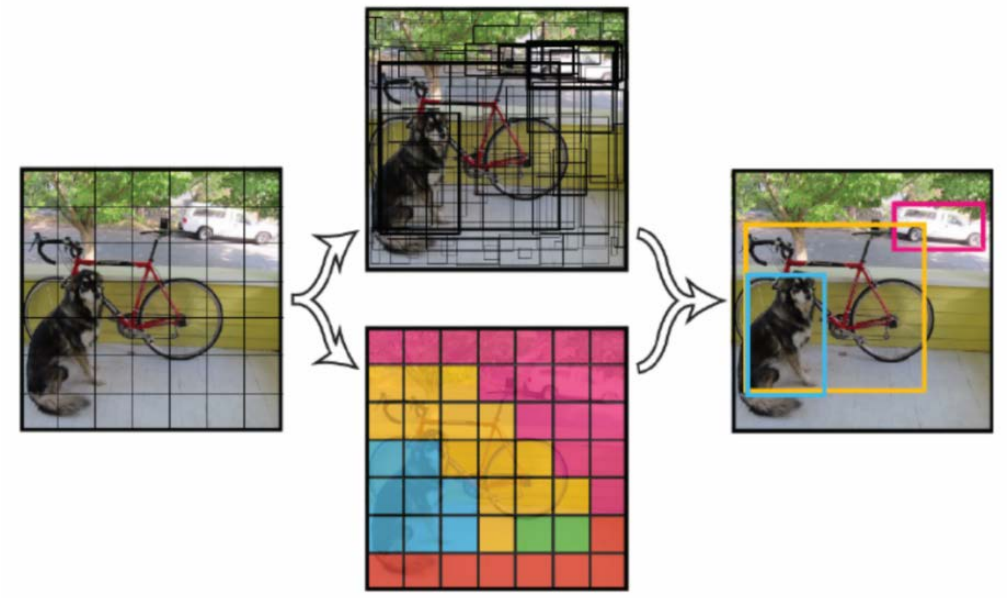


图 2.2 YOLO grid 示意图 [12]

2.3 YOLOv5 算法

2015 年，YOLO 算法的发布标志着目标检测领域的一次革命性突破。相较于传统的两阶段分类问题，YOLO 将目标检测简化为单一回归问题。Joseph Redmon 及其团队的创新之处在于他们构建了一个单一的神经网络，能够直接从输入图像的像素中预测出多个边界框和这些框的类别概率。这种设计不仅使得检测速度大幅提升，同时也减少了复杂的流程和计算成本，因此取名 YOLO。经过全球多个团队的不懈努力和持续改进，YOLO 系列陆续推出了 YOLOv2、YOLOv3、YOLOv4 和 YOLOv5，每一版都是对前一版的进一步完善和创新。YOLOv5 与之前的版本有着许多变化，首先，不再基于 Darknet[46]框架实现，而是转换为 PyTorch 框架，这一转变使得模型更易于使用和扩展。其次，YOLOv5 引入了自适应锚框机制，通过将锚框的选择集成到模型中，使得算法能够动态地学习到最适合输入数据集的锚框。这一创新提高了模型的性能和泛化能力，使其在各种不同场景下都能取得更好的检测结果。YOLOv5 有四个版本分别是 YOLOv5s、YOLOv5m、YOLOv5l、YOLOv5x。为了方便在嵌入式平台部署且满足快速检测的要求，本章选择模型参数最少的 YOLOv5n，网络结构如下。

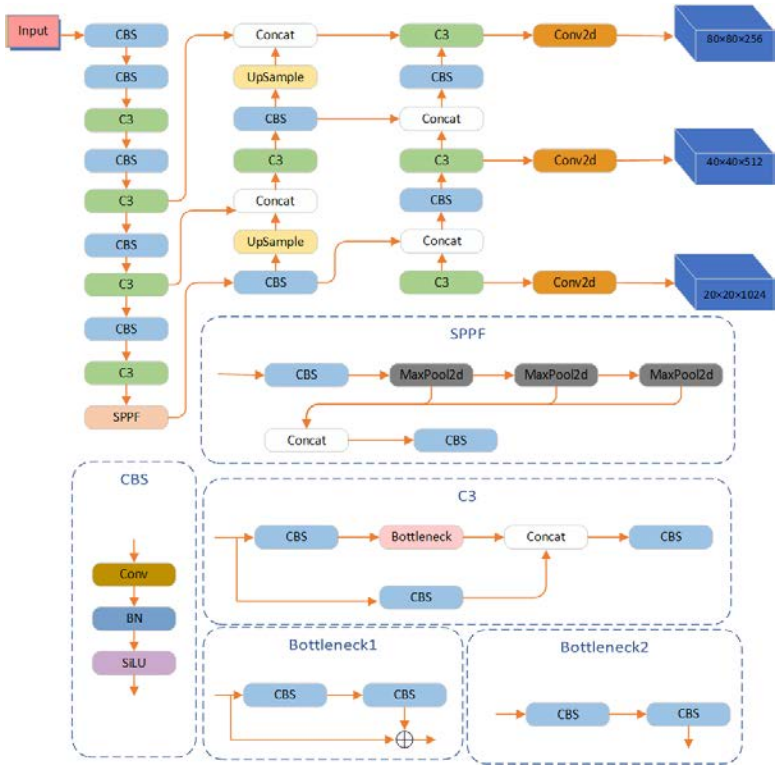


图 2.3 YOLOv5n 神经网络结构

算法神经网络架构由四个组件是构成：输入端、骨干网络、检测头和颈部。骨干网络是一个卷积神经网络结构，负责在不同图像上提取特征，这些特征将会用于后续的物体检测任务。其次，检测头负责在基于这些特征的基础上生成预测结果，包括边界框和类比预测。为了进一步优化特征提取效果，通常在骨干网络和检测头之间添加一些层，形成颈部结构。

输入端负责接收和准备图像以进行处理。一般而言就是执行图像的准备阶段任务，比如对图像进行缩放、数据增强等。

YOLOv5 的骨干网络主要由 Focus[47]结构、CBL 模块、CSP 模块与 SPP 空间金字塔池化层构成。其中 Focus 结构要通过对输入图像进行操作得到信息完整的下采样特征图，如图 2.4 所示，将 4*4*3 的原始图片经过切片与拼接之后得到 2*2*12 的特征映射。

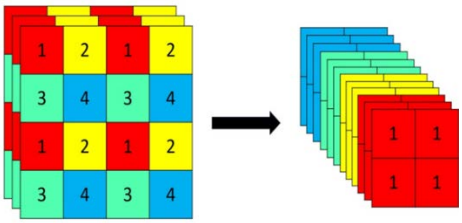


图 2.4 Focus 切片操作示意图

颈部主要采用 PANet[48]网络结构来进一步处理主干提取的特征。PANet 结构使用了自下而上的特征融合通道，这样将深层与浅层的信息进行累加，使得浅层的信息也能传递到深层中去，最终提高了不同尺度的特征表达能力。

头部负责生成最终输出，即对象检测预测。输出层预测特征图中每个网格单元的类概率和边界框坐标。然后处理这些预测以产生最终的对象检测结果。

2.4 轻量化模型选择

深度学习已经广泛的应用在图像处理领域并且极大地推动了现代智能化的发展，但是传统深度学习模型通常需要强大的计算资源和存储空间，这极大限制了嵌入式平台智能化的发展。轻量级深度学习模型的设计与优化成为了研究的热点，旨在将深度学习技术有效部署到嵌入式系统中，以实现智能化设备的广泛应用。轻量级深度学习模型的思路是通过减少算法模型参数来降低计算复杂度，目前主流方案有模型剪枝、知识蒸馏[49]、网络结构优化和量化[50]。模型剪枝是指在不影响模型性能的前提下通过移除深度学习模型中的冗余参数，减少模型的大小和计算量。知识蒸馏是指通过将一个大型复杂模型的知识转移到一个小型模型中，使得轻量级模型能够模拟复杂模型的行为。网络结构优化是指设计特殊的网络结构，通过减少参数和操作来降低模型的复杂度。量化是指将模型中的浮点数参数转换为低位宽的定点数或二进制数，以减少模型的大小和加速计算过程。相比较这几种方法，设计特征的网格结构，对嵌入式设备上部署来说更便利，更容易实现。

2.4.1 ShuffleNet 网络

ShuffleNet[51]网络是旷视科技推出的一种轻度化神经网络，同样是一种着重

应用于移动端的轻量化网络。ShuffleNet V1 中最主要的思想有两点，一是逐点分组卷积，二是通道重排，这两个操作在极大减少计算量的同时保证了良好的精度。顾名思义，逐点分组卷积就是将分组卷积和逐点卷积相结合，卷积核大小为 1×1 的分组卷积可以有效的降低模型的计算复杂度。通道重排操作使得不同的特征通道之间可以互相信息传递，极大增强了网络的特征提取能力。

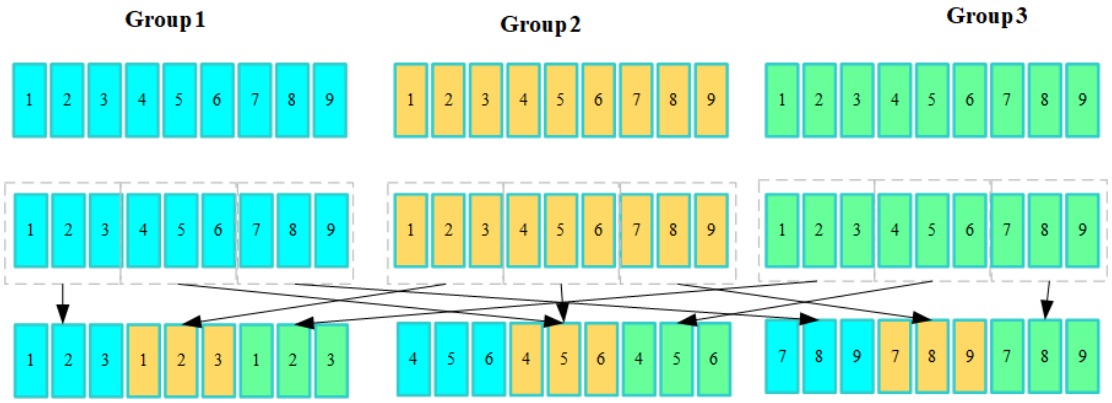


图 2.5 channel shuffle 操作

通道重组

ShuffleNet V2 的作者提出了一个与之前不同的观点，不再单单只使用浮点计算量作为间接计算网络复杂度的手段，而是使用像运行速度这类直接度量。除此之外，还需要考虑内存访问成本等因素。以这个为基准提出四个有效的网络设计原则。

2.4.2 MobileNet 网络

MobilNet[52]网络模型是谷歌公司在 2017 年提出的轻量级 CNN 网络，这个模型专注手机等嵌入式设备服务，在可承受的准确度下降范围内尽可能减少模型的参数和计算量。MobileNet V1 就是将传统的卷积替换为深度可分离卷积。针对 MobileNet V1 模型过于简单等问题，随后谷歌公司在 2018 年又提出了 MobileNetV2，相比于前一代版本，改进的地方是采用了倒残差结构和结构的最后一层使用线性层。后续又推出了 MobilNet V3[53]，在 MobilNet V2 的基础上，进行了几项关键的改进。首先，引入了 5×5 深度卷积，以增强模型的特征提取能力。其次，对 SENet 模块进行了优化，使得模型能够更有效地聚焦于关键特征。最后，为了提高模型的非线性表达能力，使用 h-swish 激活函数替代了传统的

ReLU 激活函数。这些改进共同提升了模型的性能。

2.4.3 GhostNet 网络

华为新出了一个轻量级网络，命名为 GhostNet[54]。在 GhostNet 中，作者使用一些计算量更低的操作去生成冗余的特征图，这种方法能够在维持出色检测性能的同时，显著减少模型的参数数量，进而提升模型的执行速度。这样的改进使得模型在实际应用中更加高效，能够在保持精度的同时减少计算资源消耗。在整体上来说可以将 Ghost 卷积分成两步，第一步是先是生成一个普通的 1×1 卷积，这个卷积的作用就是类似于特征整合，生成输入特征层的特征浓缩。第二步是使用深度可分离卷积获得特征浓缩的相似特征图。

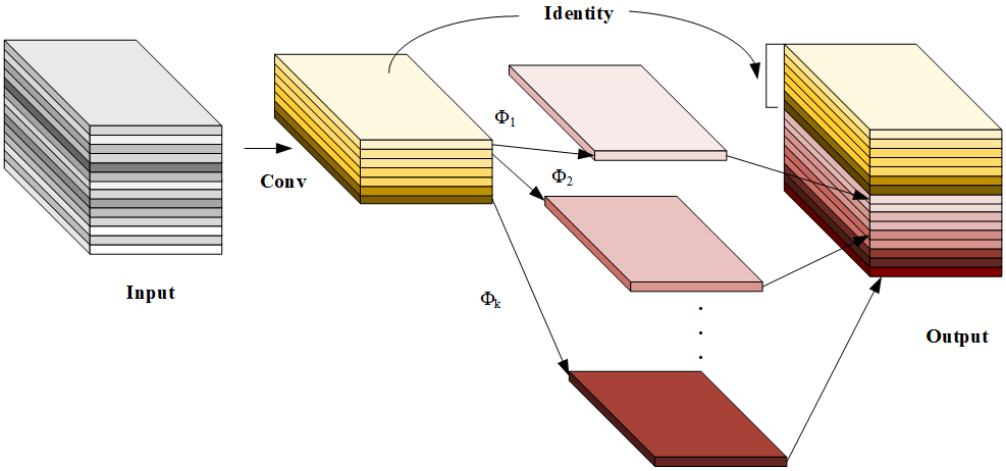


图 2.6 Ghost 网络模型

2.4.4 轻量化模型比较

在上文中分别介绍了 ShuffleNet、MobileNet、GhostNet 三类轻量化网络模型，三者各有各的优势。为了选择适合在嵌入式端进行目标检测的轻量化模型，从提取精度、计算量两个方面进行比较。本次实验选用了 ShuffleNet v2 0.5、ShuffleNet v2 1.0、ShuffleNet v2 1.5、MobileNet v2 0.5、MobileNet v2 1.0、MobileNet v2 1.4、GhostNet 0.5、GhostNet 1.0、GhostNet 1.3、MobileNetv3 small、MobileNetv3 large，为了保证实验的真实性本实验的数据来均自官方的网站。

表 2.1 轻量化网络参数对比

轻量级网络	精度	计算量
GhostNet 0.5	66.2	42
GhostNet 1.0	73.9	141
GhostNet 1.3	75.7	226
ShuffleNet v2 0.5x	60.3	41
ShuffleNet v2 1x	69.4	146
ShuffleNet v2 1.5x	72.6	299
ShuffleNet v2 2x	74.9	591
MobileNet v2 0.35	60.3	59
MobileNet v2 0.5	65.4	97
MobileNet v2 1.0	71.8	300
MobileNet v2 1.4	75.0	582
MobileNet v3 small	67.4	56
MobileNet v3 large	75.2	219

从上表中可以明显可以看出单从精度、计算量方面 GhostNet 占据优势，其他则显得稍微次之。为了更加清晰看出结果对上述表格的信息进行可视，如下图 2.7 所示。

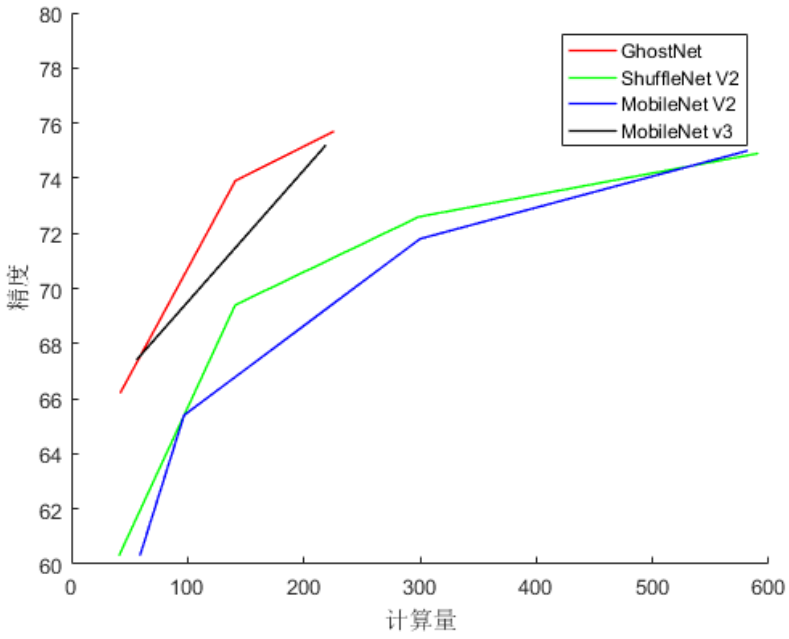


图 2.7 轻量级网络参数对比

从上图中红色代表可以发现红色的线一直占据着优势，说明从精度与计算量的角度来说，GhostNet 的性能是最好的。下面红色的线次之也就是 MobileNetv3 次排第二，而蓝色与橙色代表的 ShuffleNetv2 与 MobileNetv2 在效果上有待提高。

2.5 基于改进 YOLOv5 的轻量级目标检测算法

2.5.1 基于 GhostNet 改进网络模型

GhostNet 是一种轻量级神经网络，该网络的创建是为了解决传统深度神经网络模型的缺点，传统模型过于复杂，无法在资源贫瘠的嵌入式设备上运行。在本文研究中，对 YOLOv5 架构进行了修改，使用 C3Ghost 模块来替换原来的 C3 模块。创建该模块的目的是通过在多个卷积层之间共享权重来降低模型的计算复杂度。因此，该模型在保持准确度的同时，参数较少。这对于嵌入式实时应用来说尤为重要。除此之外，使用 Ghost 卷积模块替换了 CBS 模块。Ghost 卷积模块可以作为传统卷积层的轻量级替代品，减少模型参数的数量。这使得模型可以更好的利用计算资源，使得其更合适在资源受限的设备上。

Ghost 使用特征图之间的相关性，在不需要大量计算的前提下，生成大量特征图，如图 2.8。

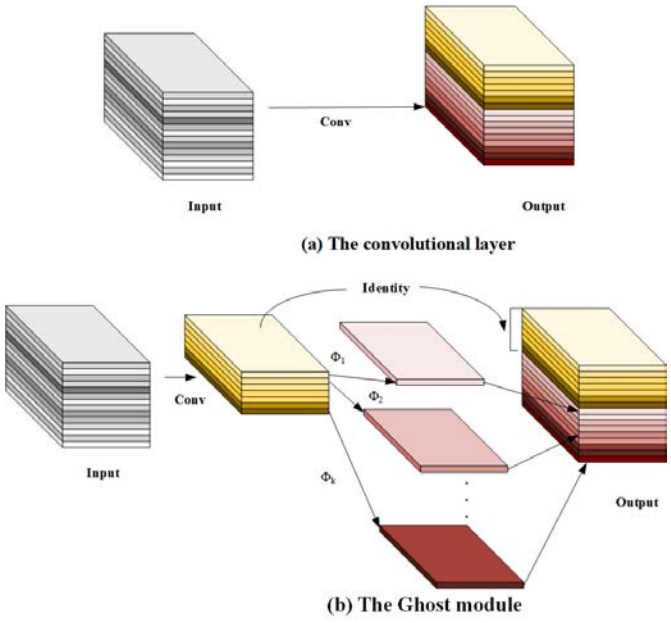


图 2.8 常规卷积与 Ghost 原理对比

对于输入图像 $X \in R^{C\times H\times W}$ 其中 C、H 和 W 分别代表了通道的数量、输入图像高度和输入图像宽度。用于生成 n 个特征图的任意卷积层的操作可以写成以下方程。

$$Y = X * f + b \tag{2.1}$$

其中，*表示卷积运算，b 表示偏差，Y 为输出，f 代表包含 n 个卷积核尺寸为 $k\times k$ 的卷积层。传统卷积中所需的计算量(Floating Point Operations, FLOPs)数量可以计算为 $n\times c\times h\times w\times k\times k$，这通常是几十万，因为滤波器的数量 n 和通道的数量 c 十分大。

Ghost 模块提出的卷积操作的运算量相比于传统卷积的运算量有了大幅的降低。假设将输入设为 $X \in R^{C\times H\times W}$，先将输入的特征向量进行常规卷积，输出的 p 个本征特征图 Y'。

$$Y' \in X * f' \tag{2.2}$$

为了获得冗余的(n-p)个特征图，将 p 个输出中的每一个本质特征图进行(q-1)次线性运算，使每张特征图都通过映射产生(q-1)个相似特征图，其中 q=n/p，最后将得到的 p 个本质特征图与经过线性运算得到的(q-1)个特征图进行堆叠，得到 n 个特征图，其线性操作可以表示为式 2.3，所以 Ghost 卷积与常规卷积运算量的比值可表示为式 2.4。

$$y'_{i,j} = \phi_{i,j}(y'_i), \forall i = 1,...,m, j = 1,...,s \tag{2.3}$$

$$r = \frac{n\times c\times h\times w\times k\times k}{p\times c\times h\times w\times k + p\times(q-1)\times h\times w\times k\times k} \tag{2.4}$$

其中 y'_i 表示 y' 中的第 i 个特征图，$\phi_{i,j}$ 中 i 表示特征图的序列号，j 表示对 i 个特征图进行的第 j 个线性运算。当线性操作的卷积核尺寸与常规卷积操作的卷积核尺寸相同是可得：

$$r = \frac{q\times c}{q + c - 1} \approx q \tag{2.5}$$

最终可得 Ghost 卷积与常规卷积模块的计算量比值的估计值 q，这个值远小于 c，故而模型具有轻量化的特点。基于此提出使用 GhostConv 与 C3Ghost 模块与 YOLOv5n 结合。

C3Ghost 模块和 YOLOv5n 中的 C3 模块相类似。主要是对 C3 模块中的一部分卷积模块加入了 Ghost Bottlenecks 模块，另一部分则继续沿用卷积、正则化

和激活函数的步骤生成特征图，最后将两部分特征图拼接得到梯度组合差异较大的特征图。

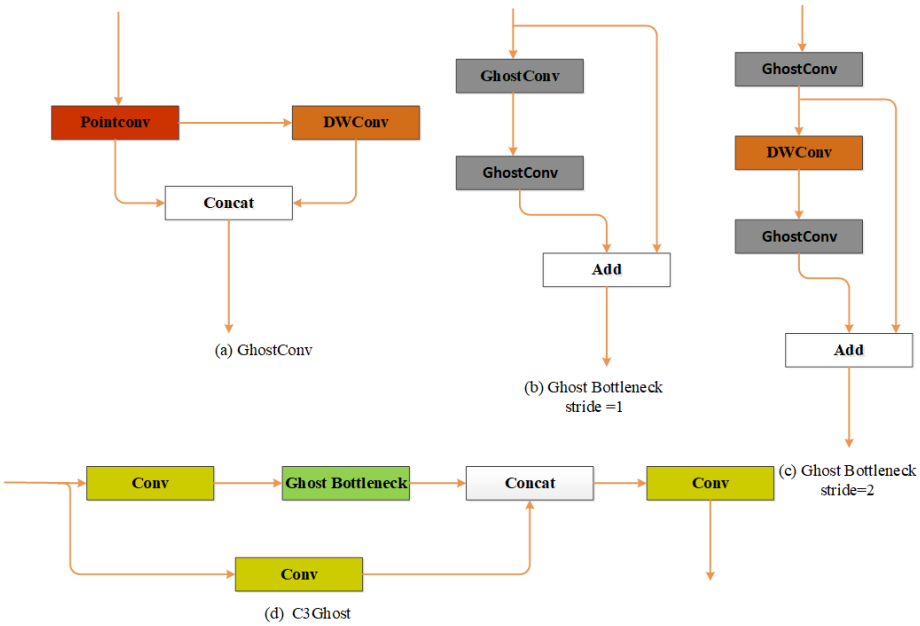


图 2.9 C3Ghost 和 GhostConv 模块

2.5.2 引入无参注意力机制的骨干网络

与通道域和空间域注意力机制不同，SimAM 模块能在不增加网络参数的前提下推断出特征图的三维注意力权重。当场景中的图像包含复杂背景以及多种类型的物体，为了准确检测到感兴趣区目标，模型需要学习能够区分对象和背景区域的判别特征。但是原始的 YOLOv5 难以解决这个问题，为此提出将 SimAM 注意力模块融入其中。通过度量目标神经元与其周围神经元之间的线性可分性，SimAM 可以识别每个通道中最具特色的神经元，这些神经元可能对应于在复杂环境中区分物体的关键信息视觉模式。SimAM 生成的 3D 注意力权重允许选择性地突出显示这些信息神经元，同时抑制不相关的神经元。这增强了 YOLOv5 从复杂图像中提取判别特征的能力。

SimAM 通过度量神经元之间的线性可分性来寻找重要的神经元，其最小能量能量函数如下所示：

$$\hat{\mu} = \frac{1}{M}\sum_{i=1}^{M} x_i \tag{2.6}$$

$$\hat{\sigma}^2 = \frac{1}{M}\sum_{i=1}^{M}(x_i - \hat{\mu})^2 \tag{2.7}$$

$$e_t^* = \frac{4(\hat{\sigma}^2 + \lambda)}{(t-\hat{\mu})^2 + 2\hat{\sigma}^2 + 2\lambda} \tag{2.8}$$

t 与 x 分别是输入特征的单个通道内的目标神经元和其他神经元，i 是空间维度上的索引号，M 是通道内神经元的数量，λ 是正则项，$\hat{\mu}$ 和 $\hat{\sigma}^2$ 表示均值和方差。从上式可以看出 e_t^* 越小，t 与 x 之间的异常就越大，也就是说神经元的重要程度越大。

$$\tilde{X} = Sigmoid(\frac{1}{E}) \odot X \tag{2.9}$$

$\tilde{X}$ 表示输出特征图，X 表示输入特征图，$Sigmoid$ 激活函数的目的是限制 E 值。

将 SimAM 注意力模块与 C3 模块相结合，组成 C3SimAM 模块然后替换主干中的最后一个 C3 模块。

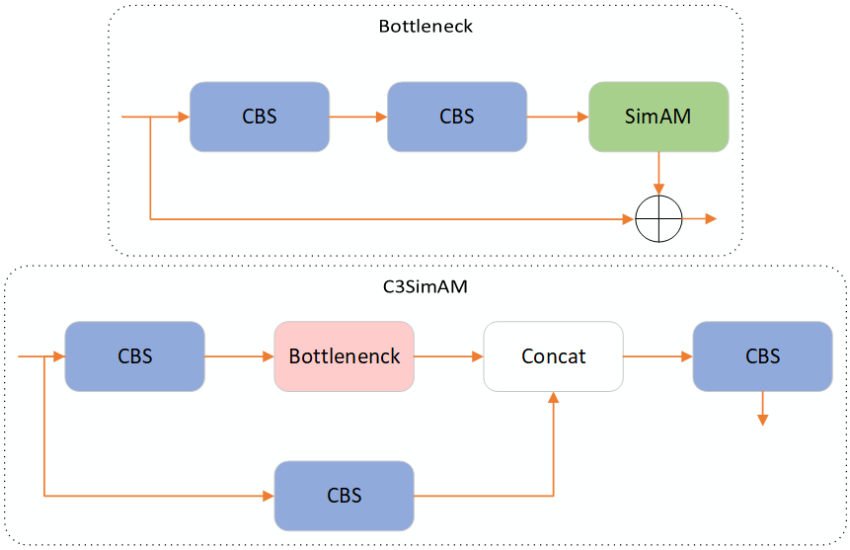


图 2.10 C3SimAM 模块

2.5.3 空间金字塔池化改进

SPP 是空间金字塔池化，是一种用于图像识别和目标检测的技术，其作用是在不同尺度下对图像进行特征提取和编码，它可以将任意大小的输入图像重新缩放到固定大小，并生成固定长度的特征向量。SPPF 将原本并行的结构改为串

行结构，相对原来的 SPP 减少了计算量。这种改进使得模型能够更高效地进行特征提取和池化操作，从而在保持准确性的同时提升了运行速度。YOLOv5 中 SPPF 的主要作用是融合多尺度特征，将同一特征图不同尺度下的特征融合到一起，丰富特征图的语义特征。YOLOv5 中原始的 SPPF 模块使用了 CBS 结构块，CBS 结构由一个 Conv2D、一个 BatchNorm2d 和激活函数构成。Conv2d 是卷积神经网络中的一种关键操作，它在二维图像上执行卷积运算，以提取图像中的特征。BatchNorm2D 是一个函数，它对输入数据的四维数组进行批量归一化处理，这能够有效地加速神经网络的训练过程，提高模型泛化能力。激活函数在神经网络中的重要作用是引入非线性因素，可以使得网络具备更强的表达能力。通过激活函数，神经网络可以学习复杂的非线性关系，从而更好地适应各种类型的数据分布和任务要求。Conv2D 是卷积首先是对输入信号进行二维卷积处理，然后将其送到 BN 层进行归一化处理，最后将结果传到激活函数 SiLU[55]。SimSPPF 采用了 CBR 结构块，使用 ReLU[56]替换原来的激活函数 SiLU。SimSPP 与 SPPF 都各有各的优点，如果考虑的是准确性，那么原始的 SPPF 模块相比于 SimSPPF 有更大的优势，因为带有 SiLU 激活函数的 CBS 结构块能够有效地捕捉多个尺度的上下文信息。如果考虑的是计算效率，带 SimSPPF 模块更可取，ReLU 激活函数的 CBS 结构，计算成本较低。

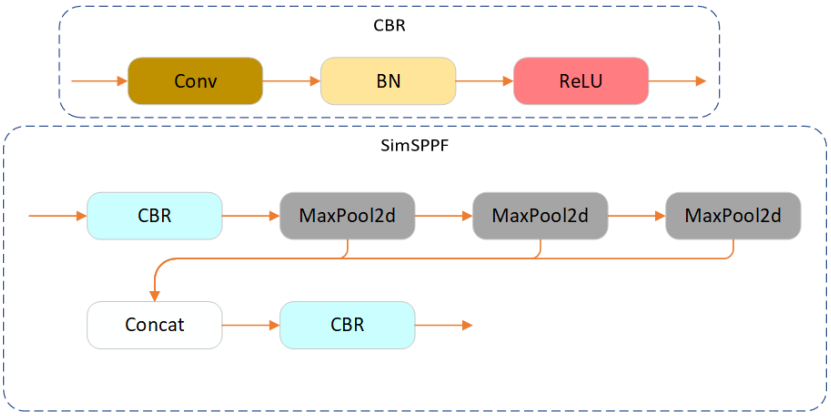


图 2.11 SimSPPF 网络结构

2.5.4 YOLOv5-SG 整体网络

如下图 2.12 所示，基于原始的 YOLOv5n 网络模型，通过集成 GhostNet 轻量级结构、SimAM 注意力机制以及 SimSPPF 模块，成功构建了一个全新的网络

模型 YOLOv5-SG。这一模型不仅在视觉目标检测方面有着高精度的特性，更通过轻量化的设计思路，显著提升了运算效率和模型部署的灵活性。具体而言，GhostNet 的引入有效减少了模型参数的数量，而 SimAM 注意力机制则帮助模型更加聚焦于关键特征，增强了特征的表达能力。同时，SimSPPF 模块的加入进一步优化了模型的特征融合能力，提升了检测的准确性。

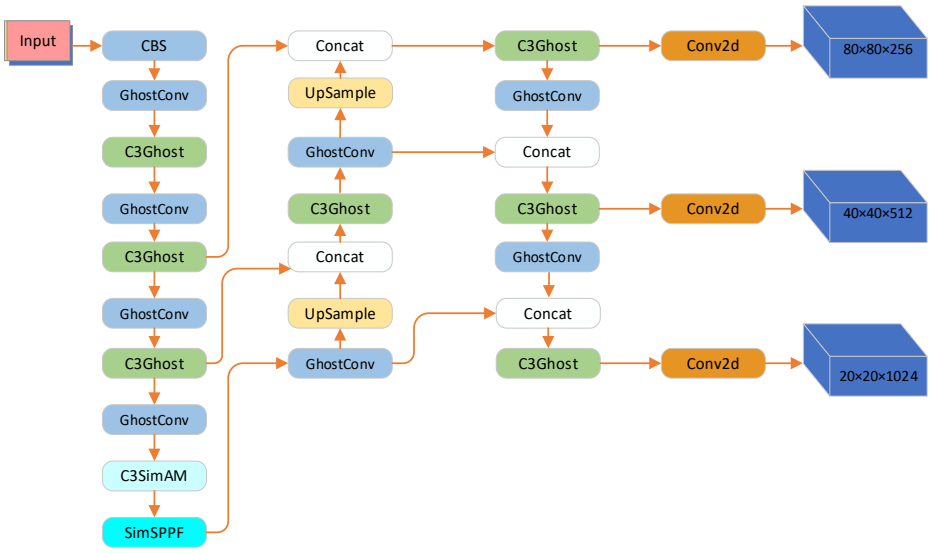


图 2.12 YOLOv5-SG 网络模型

2.6 YOLOv5-SG 算法实验验证与分析

2.6.1 数据集处理与实验设置

在目标检测中常用的开源数据集有 BDD100K、Cityscape、COCO[57]、ROAD[58]、KITTI[59]等，其中 KITTI 数据集由德国卡尔斯鲁厄理工学院和丰田美国技术研究院联合创建，以其庞大的规模和丰富的内容著称于世。数据集中包含了来自不同场景的真实图像数据，涵盖了城市、乡村和高速公路等各种不同的环境。

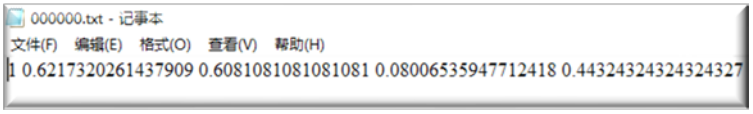
(a) 行人类　　(b) 汽车类　　(c) 骑行者

图 2.13 KITTI 数据集部分图像

本章实验采用了 KITTI 2D Object 公开数据集进行实验和评估，该数据集包括了 7481 个训练样本和 7518 个测试样本。由于 KITTI 数据集有其特有的数据格式，无法直接被 YOLOv5 识别，故而需要对其进行数据格式的转换。在 KITTI 原始数据集中存在很多的标注信息，包括检测目标的位置、遮挡、截断信息，以及检测目标在空间中的方向和置信度。首先，使用 xml 文件将 KITTI 格式数据提取出来。然后，将 xml 文件转换成 txt 文件格式。

(a) KITTI 数据原始标注格式

(b) KITTI 数据集的 xml 格式标注信息



(c) YOLO 可用数据格式

图 2.14 KITTI 数据集标注信息的不同格式

接下来还需要对标签的类别进行重新划分。KITTI 数据集的目标分为了八类，分别是汽车（car）、面包车（van）、卡车（truck）、电车（tram）、其他载具（misc）、行人（pedestrian）、坐着的人（pedestrian-sitting）以及骑车的人（cyclist）组。本文研究面向室外动态目标的目标检测算法，只以车和人作为检测对象，不关注车的具体类型和人的形态，因此本文对目标类别进行合并，将前五项融合为车（car），将行人和坐着的人融合为人（pedestrian），将骑行的人保留不变（cyclist）。

本实验批次大小设置为 16，动量设置为 0.9，初始学习率设置为 0.01，权重衰减正责项系数设置 0.005，训练迭代次数设置为 300。实验所用的软、硬件的配置信息如下表 2.2。

表 2.2 轻量化网络性能验证实验平台

实验环境	版本型号
操作系统	Ubuntu 18.04
编程语言	Python 3.8
CUDA	10.2
Pytorch	1.9.0
处理器（CPU）	Intel Xeon E5-2680 v4
显卡	NVIDIA GTX 2080Ti
显存	11G

2.6.2 评价指标

模型评估是实验分析的一个重点，经常使用的基本评价指标有精确度(Precision)、召回率(Recall)、平均准确率(Average Precision, AP)、平均准确率均值(mean Average Precision, mAP)、FPS、Param。

$$P = \frac{TP}{TP + FP} \tag{2.10}$$

P 表示预测为正的样本中有多少是真正的正样本，TP 表示把为正样本预测为正样本，FP 表示把负样本预测为正样本。

$$R = \frac{TP}{TP + FN} \tag{2.11}$$

R 表示样本中有多少正样本被正确预测了，TP 表示把正样本预测为正样本、FN 表示把正样本预测为负样本。

$$AP = \frac{1}{n}\sum_{i=1}^{n} P_i(r) \tag{2.12}$$

AP 表示单一类别的平均精度。

$$mAP = \frac{\sum_{i=1}^{k} AP_i}{k} \tag{2.13}$$

mAP 表示所有单一类别平均准确度的平均值。

在目标检测任务中，采用 *IOU* 作为判断预测框检测结果准确性的指标。*IOU* 衡量了检测框与真实框之间的重合程度。当 *IOU* 值超过设定的阈值时，认为检测结果为正样本。具体计算公式如下，*A*，*B* 分别代表了检测框和真实框面积。

$$IOU = \frac{A \cap B}{A \cup B} \tag{2.14}$$

2.6.3 算法性能分析

为了更加全面的评估模型，将原始的算法与改进后的算法在 KITTI 训练集上做测试，从训练结果所呈现的三类别 Precision-Recall 曲线围成的面积来看，检测精度在不同目标类别间存在差异。其中，检测精度最低的是行人，最高的是汽车。而且，经过优化后的模型在各类目标检测任务上的性能均得到了显著提升。具体而言，汽车平均精度提升 0.72%，行人平均精度提升 3.21%，骑行者平均提升 3.55%，整体的 mAP 提升 2.35%。

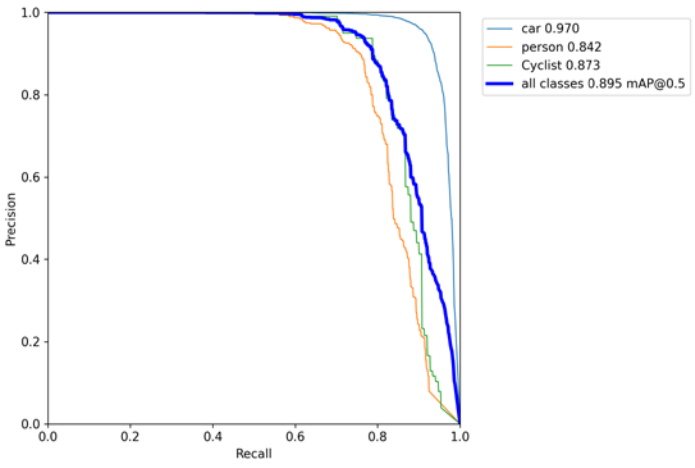


图 2.15 改进前算法 P-R 曲线图

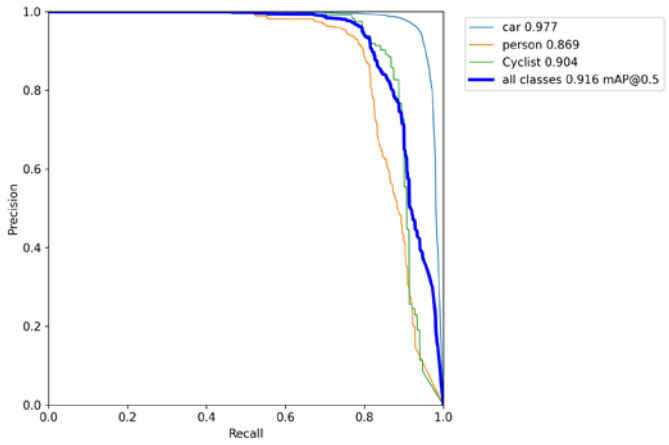


图 2.16 改进后算法 P-R 曲线图

2.6.4 消融实验

为了进一步比较改进算法模块之间的差异，对其进行消融实验。

表 2.3 消融实验

添加模块	mAP(0.5)	Params	FPS
	0.895	1.76M	109
+GhostNet	0.894	0.94M	177
+GhostNet+SimAM	0.911	0.94M	179
+SimAM+GhostNet+SimSPPF	0.916	0.94M	183

从上述表格中的消融实验可以看出 YOLOv5 在引入轻量级网络 GhostNet 后，模型的参数减少 46.6%，变为 0.94M，FPS 增长了 68 帧。而后，引入 SimAM 注意力机制后 mAP 提升 1.9%，模型的参数保持稳定，FPS 增长了 3。最后添加 SimSPPF 模块，此时模型的参数规模稳定，mAP 和 FPS 出现小幅度增长。整体而言，算法改进后模型参数减少 46.6%，mAP 提升 2.35%，FPS 增长 68%。

实验结果表明，引入 GhostNet 可以在保证精度稳定的情况下极大减少模型参数。添加 SimAM 可以在不增加模型参数的情况下增强模型的检测性能，添加 SimSPPF 可以增加检测精度和检测速度且模型参数变化不大。

2.6.5 可视化分析

为了具体分析 YOLOv5 与改进后的算法的检测效果之间的差异，接下来利用 KITTI 数据集进行测试，并将测试结果以可视化的形式展示，图中红色检测

框代表了车，橙色检测框是行人，粉红色检测框代表骑行者。

(a) 场景一

(b) 场景二

(c) 场景三

(d) 场景四

图 2.17 改进前算法检测效果

(a) 场景一

(b) 场景二

(c) 场景三

(d) 场景四

图 2.18 改进后算法检测效果

在场景一中，检测的场景较简单，改进前算法检测出了五辆汽车，改进后的算法检测精度高于改进前检测出来六辆汽车。场景二中发生了较为严重的遮挡现象，改进前算法只检测出了三辆汽车，改进后的算法检测出了五辆汽车。场景三是长距离场景，改进前算法只能检测到近处的车辆，对远处发生遮挡车辆出现较多漏检，改进后的算法虽然相比于之前有了一定性能提升，但也出现了漏检问题。场景四是一个拥挤场景，改进前算法出现两个错误，一个是行人漏检，另一个是将独立的人和自行车检测为两个骑行者。改进后算法避免了行人漏检问题，同时对于骑行者的误检行为有了改善。

根据上面四个场景检测结果可以知道，改进算法后，检测精度提高且减少了因为遮挡导致特的漏检行为，在不同的场景中有着良好的普适性和泛化性。但是改进后的算法，在复杂环境下也会出现漏检、误检行为，为了进一步算法的性能后续将使用融合多种传感器的方法进行目标检测。

2.7 本章小结

在本章节中，主要聚焦于目标检测算法的优化，针对传统深度学习模型结构复杂、计算量庞大以及硬件需求较高等问题进行了探讨。通过引入轻量级模块 GhostNet，从而有效降低了模型的参数量和网络深度。这一调整不仅使得模型更为精简，使其在资源受限的环境下更具可行性。然而，仅凭简化模型结构并不足以保证检测精度的提升。因此，为了进一步优化检测性能，引入了 SimAM 注意力机制。通过引入这一机制，模型能够更加准确地聚焦于关键区域，提高目标检测的精度和鲁棒性。此外，引入了 SimSPPF 模块。SimSPPF 模块在保留 SPPF 模块多尺度信息融合的基础上，通过引入相似性度量机制，进一步提高了模型的实时性能。通过 KITTI 公共数据的验证，mAP 达到了 91.6%，比原始模型提升了 2.35%，且模型参数量下降 46.6%。

第3章 基于改进 PointPillars 的激光点云目标检测

3.1 引言

在上一章提出使用基于深度学习的视觉目标检测算法对物体进行检测，但是从实验结果可以看出使用相机进行目标检测，容易受遮挡、光线等环境影响。基于深度学习的激光点云目标检测算法是近年来在机器人感知领域备受关注的技术。该算法利用深度学习模型对三维激光点云数据进行处理和分析，实现对目标物体的准确检测。然而，由于激光雷达的激光束在远距离时会发散和衰减，小物体产生的点云数据过于稀少易被背景噪声掩盖，使得算法很难对其准确识别和定位。针对这个问题，提出一种基于改进 PointPillars 的激光点云目标检测算法。首先，二维特征提取网络中引入 ECA 注意力机制增强对伪图像位置特征信息的提取。然后，为了处理包含负数的点云数据，引入了 Softplus 激活函数。最后，优化目标朝向损失函数提高模型的精确度和鲁棒性。此外，为了减轻嵌入式平台的计算负担，采用了体素滤波的方法对原始激光点云数据进行预处理。

3.2 基于深度学习的点云检测算法

传统点云检测算法在特征提取与模型设计方面高度依赖于人工经验，这导致其泛化能力相对有限。特别是在处理复杂场景和多变目标时，传统算法往往难以适应，其检测性大受影响。基于深度学习的方法能够自动学习和提取深层次的特征表示，在处理复杂场景时能够展现出更优越的性能。然而，点云数据由于其稀疏性、无序性、非结构化和分布不均匀的特性，不适合直接应用图像领域的经典深度学习神经网络进行处理。点云的表征方式决定了如何从点云数据中提取表达特征，进而支持后续的三维检测任务。根据特征提取方式的不同，基于深度学习的点云检测算法可以分为三类：基于体素的方法、基于投影的方法和基于点的方法。

3.2.1 基于体素的方法

基于体素的方法将不规则的点云数据转化为规则的纯矩阵形式，通过离散化 3D 空间为固定大小的体素栅格，从而适应卷积操作。这种方法能够极大保留点云空间信息。然而稀疏点云数据分布不均，大量区域缺乏有效的点云数据，导致在体素化过程中生成大量的空体素。这些空体素不仅增加了数据处理的复杂性，还导致计算资源的浪费。

体素法的经典网络 VoxelNet 通过 VFE 层来有效地处理点云数据，如下图 3.1。通过将逐点特征与局部聚集特征相结合，实现了体素内的点间交互。该算法堆叠多个 VFE 层允许学习用于表征局部 3D 形状信息的复杂特征。具体而言，VoxelNet 将点云划分为等间距的 3D 体素，通过堆叠的 VFE 层对每个体素进行编码，然后通过一个中间卷积层，扩大感受野的同时进一步获得更多的特征信息，最后借助一个 RPN 模块，类别分支输出检测物体的类别信息、回归分支输出中心点、长宽高以及角度相对于 Anchor 的偏移量。

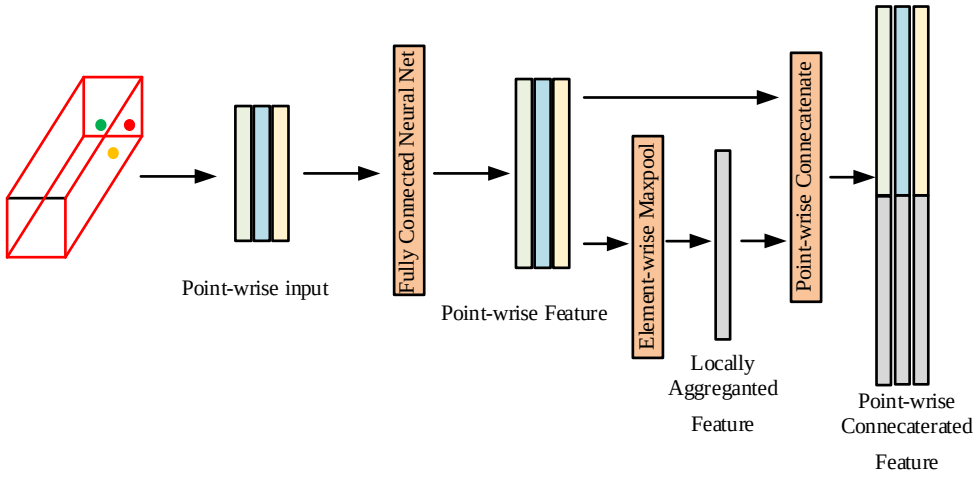


图 3.1 体素特征编码

3.2.2 基于点的方法

在将点云数据转换为投影或体素表示的过程中，空间信息的丢失是一个不可避免的问题。因此，直接处理原始点云对于保留这些重要的空间信息显得尤为关键。代表性的网络有 PointNet，网络结构如图 3.2 所示。PointNet 直接对点云进行处理，对输入点云中的每一个点，学习其对应的空间编码，之后再利用所有

点的特征得到一个全局的点云特征。PointNet 由分类网络和分割网络两部分组成。分割网络以 n 个点作为输入，应用输入和特征转换，然后通过最大池化来聚合点特征，输出 k 个类的分类分数。分割网络是分类网络的扩展，该网络连接全局和局部特征，并输出每个点的分数。多层感知机(Multilayer Perceptron, MLP)由 5 个隐藏层组成，神经元大小分别为 64，64，64，128，1024，所有点共享一个 MLP 副本。

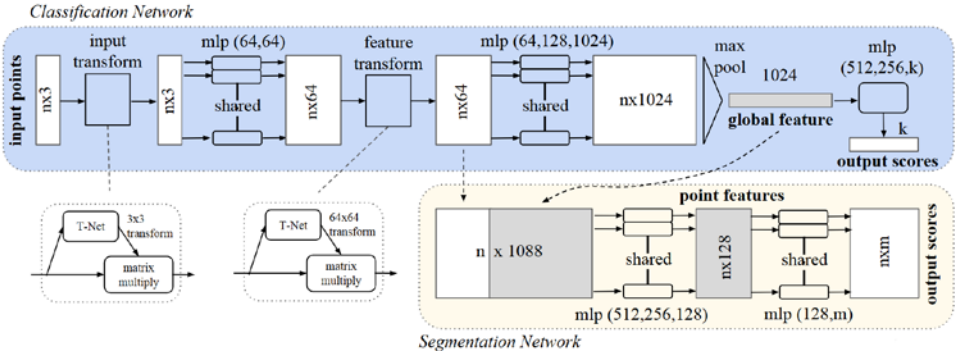


图 3.2 PointNet 网络结构[60]

3.2.3 基于投影的方法

在三维点云处理中基于投影的方法是一种常用的特征化手段。这种方法的核心思想是将三维的点云数据投影到二维平面上，一般是俯视图或前视图，从而将三维空间中的目标检测问题转化为二维平面上的目标检测问题。通过投影，可以充分利用二维卷积神经网络在处理图像数据时的成熟技术和高效性能。

（1）前视图

在三维点云处理中，前视图作为一种特殊的二维投影方式，展现出类似于圆柱形图像的特点。通过将三维空间的点云数据映射到二维平面上，每个像素得以集成多个特征通道的信息，从而形成一个富含空间特征的前视图像。为了充分保留点云中的关键特征，设计了多种统计前视特征，旨在通过这些特征提高目标定位和分类的精度。前视投影方法能够有效地将三维点云数据压缩为紧凑的二维图像，这一特性使得现成的二维检测器能够直接应用于点云目标检测任务。在投影过程中，深度信息被精心地嵌入到前视图中，通过捕捉不同视图间的依赖关系，显著提升了目标检测的性能。这种方法不仅简化了数据处理流程，还提高了目标检测的效率和准确性。

然而，前视图投影方法也面临着一系列挑战。其中，透视性、尺度变化和遮挡等问题带来的信息损失尤为突出。透视性导致远处物体的特征在投影过程中被压缩，造成信息损失；尺度变化则使得不同大小的物体在投影后的图像中难以区分；而遮挡问题则进一步加剧信息损失。这些难点限制了前视图投影方法在处理复杂场景时的性能。

（2）俯视图

在三维点云处理中，俯视图投影作为一种将三维数据映射到二维平面的重要技术，通过对 Z 方向的高度信息进行编码操作，如转化为高度差、平均高度等统计量，以在二维空间中体现三维空间的特性。为了弥补降维过程中的信息损失，通常将点云的密度和强度信息作为附加的特征通道叠加到投影图像中，以增强二维图像的特征表示能力。然而，这种方法在很大程度上依赖于二维检测算法的性能，并且需要精心设计人工特征以在二维投影中保持三维空间特征。此外，投影的分辨率和特征通道的数量直接影响计算效率。高分辨率和更多的特征通道能够提供更丰富的信息，但同时也会增加计算负担，降低处理速度。

3.3 PointPillars 算法

在深度学习点云目标检测中基于体素的方法可以极大保留点云数据的完整性，但是传统基于体素的算法计算量过于复杂。PointPillars 算法采用点柱特征提取方法，能够高效地从点云中提取形状、位置和方向等信息，并将其转换为高效的 3D 特征表示。这种表示方式使得算法能够在保持较高的检测精度的同时，保持较低的计算成本，提高算法的实时性能。该算法主要分成三个阶段：

（1）将点云数据通过点柱特征编码网络生成点云伪图像。

（2）使用二维卷积特征提取网络进行特征提取同时保留点云的 3D 信息。

（3）通过 SSD 算法使用特征图生成当前尺度的锚框，并将其传递到检测网络进行方向估计和类别分类。

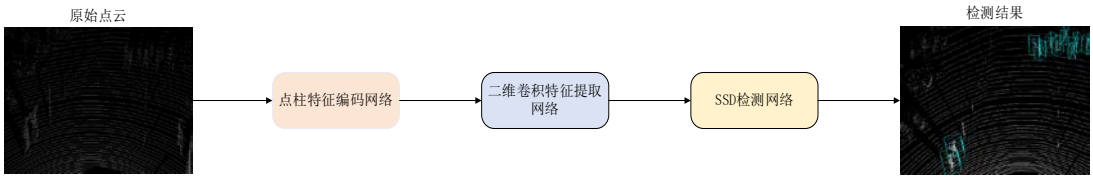


图 3.3 PointPillars 算法流程

3.3.1 点柱特征编码网络

该部分网络结构的作用是将点云转换为伪图像，转换过程如图 3.4 所示。

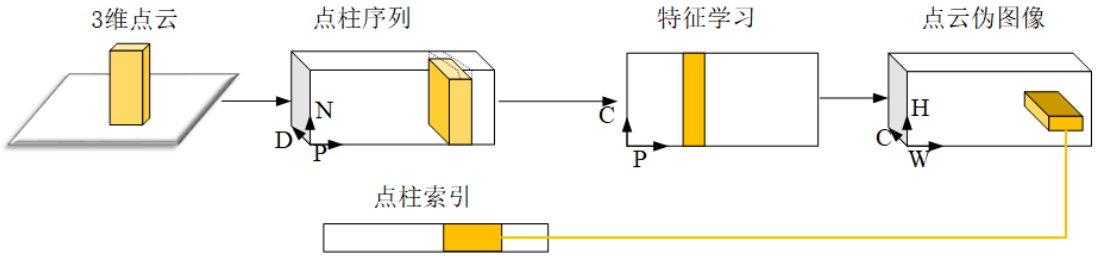


图 3.4 点柱状特征编码网络

（1）首先，输入点云被分割成多个 Pillar 单元，然后每个 Pillar 中的每个点云被编码成一个 9 维的向量 $\mathrm{D}(x,y,z,r,x_c,y_c,z_c,x_p,y_p)$。其中 x， y， z， r 分别表示点云在三维空间中的 3 个坐标和反射强度；x_c， y_c， z_c 表示到该 Pillar 中所有点的算术平均值点的偏移量；x_p， y_p 表示该点到该 Pillar 的中心的偏移值。

（2）由于点云数据的稀疏性，可能很多 Pillar 都不含点云或者包含的点云数量比较少，考虑到计算复杂度的问题，会对 Pillar 的数量进行限制，最多处理 P 个非空的 Pillar，同时每个 Pillar 中最多包含 D 个点云特征向量，如果点云数大于 N，则采用随机采样的方法从中选取 N 个，反之，如果点云的数量少于 N，则用零填充的方法填充到 N 个。通过上述方法，就将一帧点云数据编码成了一个维度为 (D,P,N) 的稠密张量。

（3）每个包含 D 维特征的点分别通过线性层、BatchNorm、ReLU 激活函数处理后，生成维度为 (C,P,N) 的张量；然后对每个 Pillar 单元进行最大池化操作，输出一个维度为 的张量。

（4）最后一步是通过一个 scatter 算子生成伪图像。通过每个点的 Pillar 索引值将上一步生成的 (C,P) 张量转换回其原始的 Pillar 坐标用来创建大小为 (C,H,W) 的伪图像。

3.3.2 二维卷积特征提取网络

二维卷积特征提取网络是一个包含两个子网络的金字塔结构。

（1）自上而下的特征提取：使用多个层次从上到下提取特征，然后逐渐减

少空间的分辨率，但是网络提取到的特征在语义上来看变得越来越丰富。

（2）上采样与级联：上采样将特征图的分辨率增大到与原始的输入图像一样大小，而后使用级联将上采样得到的特征与自上而下的特征进行联系，这样保留了更多的信息。

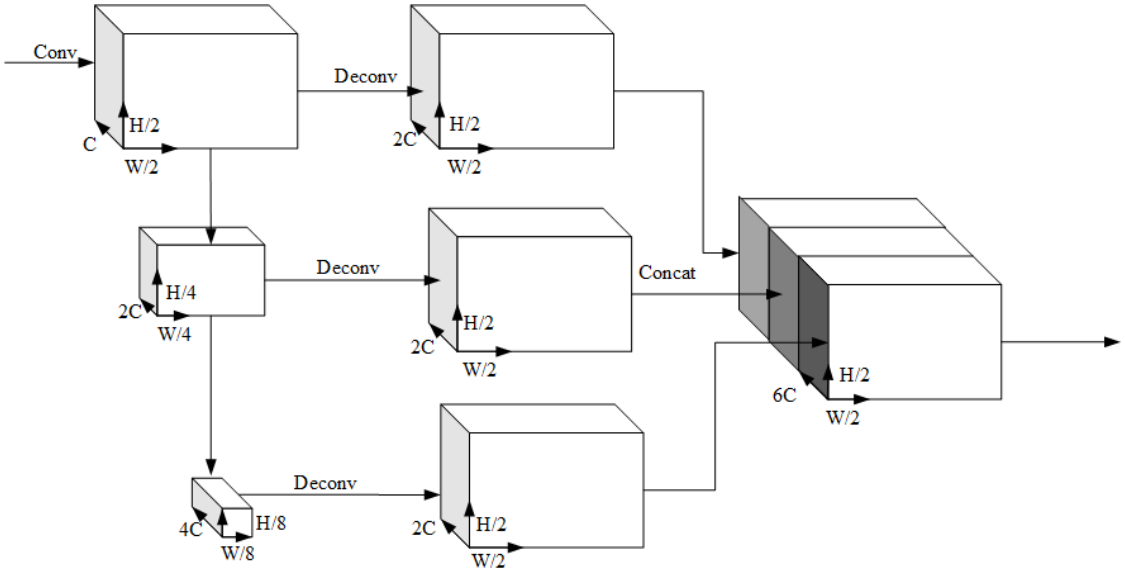


图 3.5 二维卷积特征提取网络结构图

自下而上的子网络由一系列卷积组块组成，块（S、L、F）中 S 表示步长，L 表示 3×3 大小的 2D 卷积层数，F 是输出通道数。在每个块后面有 BatchNorm 和 ReLU。为确保网络块操作在接收到步长 Sin 输入后，仍保持为 S，层内的第一个卷积步幅应设为 S/Sin。块内所有后续的卷积步长为 1。通过柱状特征编码网络有 C=64 的输出特征，所以三个块的通道数为 64，128 和 256。上采样连接网络的最终特征通过上采样和连接结合起来。首先，对特征进行上采样 Up(Sin，Sout，F)，从初始步长 Sin 到最终步长 Sout，使用 F 最终特征的转置二维卷积。接下来，BatchNorm 层和 ReLU 层被应用到上采样用于提取特征。最后将所有模块的特征组合拼接起来，作为检测模块的特征输入。

3.3.3 SSD 检测网络与损失函数

检测头采用了单步检测算法 SSD 来进行二维边界框的回归和分类，其检测速度与精度俱佳。为了更好的处理点云，SSD 网络加入了瞄点的方法来适应多尺度目物体检测任务。主干采用了 VGG16 作为基础网络，该部分主要对图像的初

步特征提取。其后，在主干之外添加了 Conv8、Conv9、Conv10 和 Conv11 卷积层，用以获得更多的特征图。最后 Multi-box Layers 对最终的目标分类检测以及非极大抑制[61]的回归定位操作。特征提取和体素分割后的三维点云张量经过 SSD 算法的一系列操作转换成边界框预测。由于生成了大量的边界预测框，下面要进行分类操作来确定每个边界框中是否存在目标，并且进行非极大抑制用以去除冗余信息。

Pointpillas 网络的总损失函数由多部分组成：

$$L=\frac{1}{N_{\text{pos}}}(\beta_{loc}L_{\text{loc}}+\beta_{\text{cls}}L_{\text{cls}}+\beta_{\text{dir}}L_{\text{dir}}) \tag{3.1}$$

N_{pos} 表示大于指定的 IOU 阈值的框的数量，L_{loc} 表示定位回归损失函数，L_{cls} 表示分类损失函数，L_{dir} 表示方向损失函数，方向损失函数在离散方向上使用 Softmax 函数。β_{loc} 取 2，β_{cls} 取 1，β_{dir} 取 0.2。

检测框定位回归损失函数 L_{loc} 采用 Smooth L1 函数，可以有效防止梯度爆炸。

$$\begin{aligned}&\Delta x=\frac{x_{gt}-x_p}{d_p},\ \Delta y=\frac{y_{gt}-y_p}{d_p},\ \Delta z=\frac{z_{gt}-z_p}{d_p}\\&\Delta w=\log\frac{w_{gt}}{w_p},\ \Delta l=\log\frac{l_{gt}}{l_p},\ \Delta h=\log\frac{h_{gt}}{h_p}\\&\Delta\theta=\sin(\theta_{gt}-\theta_p)\end{aligned} \tag{3.2}$$

最终的定位损失函数 L_{loc} 如式 3.3 所示。

$$L_{loc}=\sum_{b\in(x,y,z,w,l,h,\theta)}Smooth_{L1}(\Delta b) \tag{3.3}$$

Δ 表示真实框与预测框之间的差值，下标 p 为预测框所有，下标 gt 为真实框所有。b 为预测值与真实值的差值，Δb 为定位回归残差。系数 β 通常设置为 1。分类损失函数 L_{cls} 采用 Focal 损失函数，能动态地调整难易样本的权重，使得模型在训练过程中更加关注那些难以分类的样本。

目标分类损失函数计算公式如式 3.4 所示。

$$L_{cls}=-\alpha_a(1-p^a)^{\gamma}\log p^a \tag{3.4}$$

α_a 为调制因子，p^a 为 anchor 的类别概率。γ 为常系数，其值为 2。

3.4 基于改进 PointPillars 激光点云目标检测算法

VolexNet 算法作为体素法的经典算法，适用于多种场景和目标类型。无论是室内还是室外环境，无论是静态还是动态目标。但是传统体素法计算量过大不适合平台部署。PointPillars 算法相对其他三维目标检测算法选择使用点云立柱化，将点云转成为了二维的伪图像，极大加快了检测速度。但是由于目标越小，激光点云反射的有效点越是少，导致小物体检测不精确。在道路上机器人利用点云进行检测的时候很容易将行人、小树苗等小型障碍物识别出错，造成严重的误检、漏检行为。针对小目标检测问题，做出以下三点改进：

（1）ECA 注意力机制与特征提取网络结合，增强网络特征其提取能力。

（2）引入 Softplus 激活函数来增强对有效信息的处理能力。

（3）Softmax 损失对于角度回归不是最佳选择，优化目标朝向损失函数，提高检测的精度。

3.4.1 融合注意力机制的特征提取网络

PointPillar 算法通过二维卷积特征提取网络提高检测速度，但是对小物体的性能不佳，本文将注意力机制引入二维特征提取网络中。注意力机制的核心思想是让模型更加关注重要的特征，忽略不重要的特征，从而提高模型的性能。在二维特征提取网络中引入注意力机制，可以帮助模型更好地关注小物体的特征，从而提高对小物体的检测性能

注意力机制是受人类行为特征启发而产生的一个概念，即在决策时选择性地使用数据中的重要部分，而不是平等地对待所有信息。在注意力机制中，数据点的表示向量被加权并与之对应的权重向量相加，得到一个带权重的表示向量。通过这种方式，注意力机制使得网络能够学习到伪图像中的重要信息，抑制不重要的信息。由于参数数量少、可解释性高、鲁棒性强等优点使其广受欢迎。

ECA-Net 是视觉模型中经常使用的一种通道注意力模块，具有即插即用的支持，允许输入特征图在通道方向上得到增强，而最终的输出不会影响输入特征图的大小。与 SENet 相比，ECA 注意力机制将原来通过全连接层学习通道注意力信息替换为 1×1×1 卷积来捕捉不同通道之间的信息，减少了参数数量，消除了 SENet 对通道注意力预测的负面影响，避免了同时获取所有通道之间关系的

低效和不必要性。

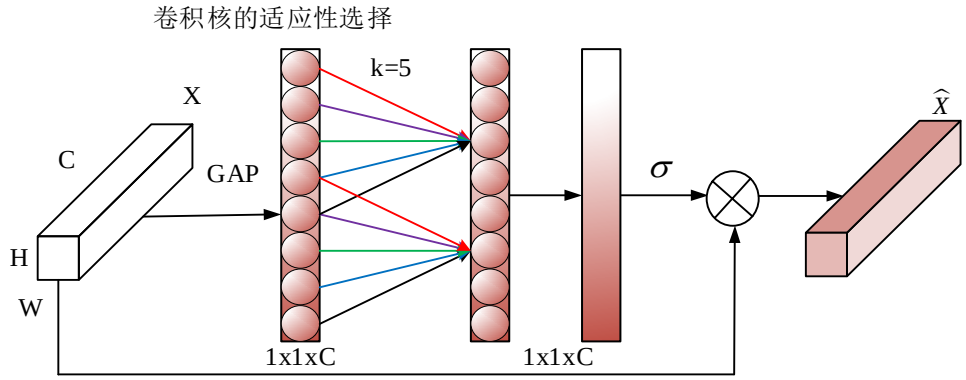


图 3.6 ECA 注意力机制

ECANet 的主要思想是摒弃原来 SE 模块的中的全连接层，利用一维卷积代替经过全局平均池化(Global Average Pooling)后的特征通过卷积核大小为的一维卷积进行学习，卷积核大小 k 表示该通道附近有多少邻居参与该信道的预测即局部跨信道交互的覆盖率，考虑每个通道及其 k 个邻居来捕获跨通道交互信息。由于 ECA 注意力机制使用到一维卷积，那么卷积核大小 k 的选择就十分重要，为了避免手动优化参数 k，ECANet 提出自适应地确定 k，卷积核大小的计算如式 3.5 和 3.6。

$$C=\phi(k)=2^{(\gamma*k-b)} \tag{3.5}$$

$$k=\psi(C)=\left|\frac{\log_2(C)}{\gamma}+\frac{b}{\gamma}\right|_{odd} \tag{3.6}$$

经过一维卷积后，ECANet 注意力机制学习到的通道注意力用 $W(k)$ 表示，$W(k)$ 中只涉及 $k\times c$ 个参数，参数量小于 SENet，而且避免了不同局部跨通道交互信息的完全相互独立，$W(k)$ 计算公式如式 3.7。

$$W(k)=\begin{bmatrix} w^{1,1} & \cdots & w^{1,k} & 0 & 0 & \cdots & \cdots & 0 \\ 0 & w^{2,2} & \cdots & w^{2,k+1} & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & w^{c,c-k+1} & \cdots & w^{c,c} \end{bmatrix} \tag{3.7}$$

ECANet 中所有通道共享权重的计算公式如式 3.8 和 3.9 所示，式中 y_i 表示通道，w_i 是通道 y_i 的权重，φ_i^k 表示与通道 y_i 相邻的 k 个通道，$C1D_k$ 表示卷积核

为 k 的一维卷积操作。

$$w^i = \sigma(\sum_{j=1}^{k} w_i^j y_i^j), y_i^j \in \varphi_i^k \tag{3.8}$$

$$w = \sigma(CID_k(y)) \tag{3.9}$$

ECA 模块的使用，可以使网络聚焦于伪图像特征中的重要信息，忽略无关信息，从而提升模型的表达力，且不会对算法的训练、运行、计算和存储等性能造成过多的额外负担。对于 PointPillars 算法，由于点云的列划分和随机采样，存在信息丢失的问题。为此，提出在二维卷积特征提取网络中降采样模块 Block1、Block2、Block3 后面依次连接 ECA 模块，重构二维特征提取网络，以解决 PointPillars 由于列划分导致点云空间丢失，从而影响目标检测精度的问题。ECA 注意力机制自适应地加强重要特征，抑制列通道内部的无关特征，以弥补空间信息的丢失。

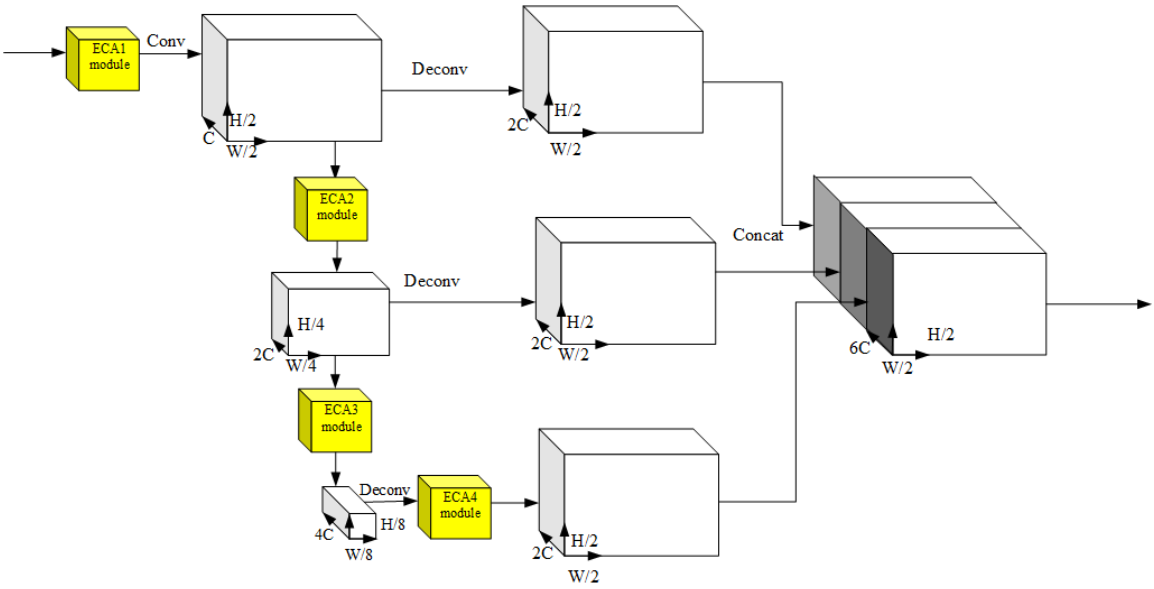


图 3.7 引入 ECA 注意力机制的二维卷积特征提取网络

3.4.2 优化损失函数

在 PointPillars 算法中，使用 Softmax 分类损失函数学习物体的方向，但是 Softmax 损失对于角度回归问题不是最优的选择，在面对多个类别的情况会存在限制。

为了改进目标朝向损失函数，使用 Smooth L1 损失函数，将目标朝向的估计看成是一个回归问题，从而增加检测精度和鲁棒性。

$$SmoothL_1 = \begin{cases} 0.5x^2 & ,|x|<1 \\ |x|-0.5 & ,others \end{cases} \tag{3.10}$$

引入余弦相似度来度量预测角度与真实角度之间的相似度，可以更好地处理角度的周期性。余弦相似度可以表示为：

$$L_\theta = \cos\theta_i \cdot \cos\theta_i' + \sin\theta_i \cdot \sin\theta_i' \tag{3.11}$$

θ_i 是实际的角度，θ_i' 是预测的角度。

目标朝向损失函数计算公式如式 3.12。

$$L_{Dir} = \frac{1}{N_{pos}} \sum_{i=1}^{N_{pos}} SmoothL_1(L_\theta - 1) \tag{3.12}$$

3.4.3 Softplus 激活函数

在进行检测的时候目标的位置、方向、大小和其他信息可能包含负值。由于 ReLU 函数的输出在输入为负时为零，因此这些负值的重要信息可能会丢失，而 Softplus 函数可以为负输入提供非零输出，并更好地捕获负值的信息。同时，Softplus 函数也具有非线性特性，可以对输入进行更丰富的非线性变换。通过引入 Softplus 函数，可以提高识别目标的准确性。对于小目标检测任务，小目标通常具有更微妙的特征和形状变化，Softplus 功能可以更好地捕捉这些细节。与 ReLU 函数相比，Softplus 函数具有更好的梯度传播特性，可以更有效地传递梯度信息。

$$Softplus(x) = \log(1+e^x) \tag{3.13}$$

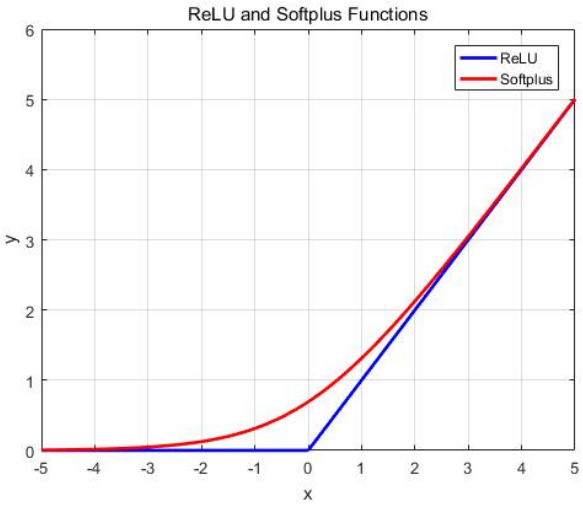


图 3.8 Softplus 函数与 ReLU 函数对比图

3.5 改进的 PointPillars 算法实验验证与分析

3.5.1 算法验证平台

为了验证改进算法，基于 OpenPCDet 搭建了开发环境，利用这框架对改进算法进行实验。该实验使用 Pytorch 框架实现网络结构，使用 GPU 进行训练和测试。Adam 优化器实现端到端的训练，批大小为 4，点柱的 x 和 y 分辨率设置为 0.16m，点柱数为 12000，点柱内最大点云数为 100。具体实验软硬件配置如下表 3.1 所示。本章实验汽车类别的 IOU 阈值在简单、中等和复杂情况下都被设置为 0.7，行人和骑行者类别的 IOU 在简单、中等、复杂场景下都设置为 0.5。

表 3.1 实验平台

实验环境	版本型号
操作系统	Ubuntu 18.04
编程语言	Python 3.9
CUDA	10.1
Pytorch	1.9.0
处理器（CPU）	Intel Xeon E5-2680_v4
显卡	NVIDIA GTX 2080Ti
显存	11G

3.5.2 数据集与点云预处理

（1）数据集

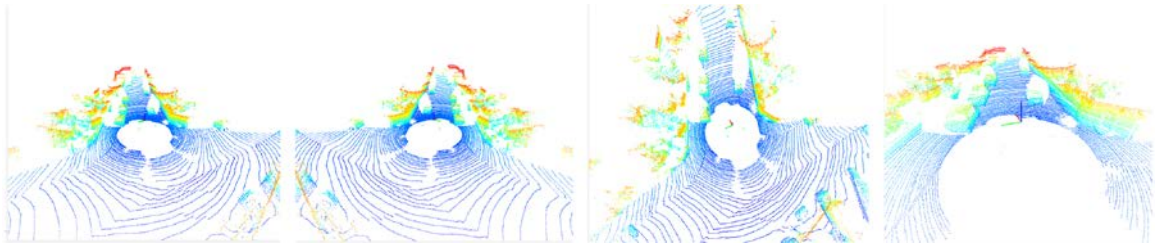
KITTI 数据集是一个从大众旅行车上采集的数据集，常用于移动机器人和自动驾驶研究。总的来说，使用各种传感器模式，记录了 6 个小时的交通场景。场景多种多样，从农村地区的高速公路到有许多静态和动态对象的市中心场景。数据经过校准、同步和时间戳，并提供校正后的原始图像序列。数据集还包含 3D 轨迹形式的对象标签，为立体、光流、对象检测和其他任务提供在线基准。

本章实验采用了 KITTI 3D Object 公开数据集进行实验和评估，该数据集包括了 7481 个训练样本和 7518 个测试样本，每个数据都包含一组的激光点云和 RGB 图像。数据集待检测对象主要由汽车、骑行者和行人类别组成，根据数据集检测目标的差异，将检测对象划分为简单、中等、困难三个等级。这种细分有助于评估算法在不同场景和难度下的性能表现，为进一步优化模型提供了重要参考。KITTI 数据采集车配有 2 个 140 万像素彩色相机和 2 个 140 万像素的灰度相机，一个 64 线的 3D 激光雷达，4 个光学镜片，以及 1 个惯性导航系统

（2）数据预处理

数据增强在点云目标检测中扮演着至关重要的角色。通过扩充训练数据集的大小和多样性，数据增强使得模型能够接触到更多的数据变体，从而更全面地学习数据的内在规律和模式。这不仅有助于提升模型的泛化能力，还有效地防止了过拟合现象的发生。在点云处理中，由于点云所构成的世界是一个具有尺度的三维空间，因此可以利用一些特定的变换来生成新的点云数据。例如，通过对点云进行镜像翻转，可以获得具有不同朝向和角度的点云样本；通过尺度缩放，可以模拟不同距离下的点云数据，使模型对尺度变化更加鲁棒。此外，沿着某个轴进行旋转或平移也是有效的点云变换方法，它们能够产生具有不同空间位置和姿态的点云数据。

这些点云变换操作不仅简单易行，而且能够显著地增加训练数据集的多样性。通过应用这些变换，我们可以获得更多不同的点云数据，从而为模型提供更为丰富的训练样本。这将有助于模型更好地学习到点云的内在特性，提升其在各种实际场景下的检测性能。



(a) 原图 (b) 镜像点云 (c) 倾斜点云 (d) 放大点云

图 3.9 三种点云数据增强方式

激光雷达每秒发射的激光脉冲信号数量，通常在数万个到数百万个之间。这样庞大的数据量，对系统的算力而言是个巨大挑战。为了确保目标检测的高效进行，采用体素网格滤波技术来精简和优化原始的激光点云数据。体素网格滤波是一种有效的降采样方法，其基于体素的概念，将点云数据分割成多个体素栅格，然后在每个栅格中计算出重心点，让重心点代替栅格中全部的点。这样一来，原始点云中的数据量得以显著减少，同时保留了关键信息，从而实现了数据的降采样。通过体素滤波，可以在保持数据结构的同时，降低数据的复杂性。

为了找到适合本章实验场景的体素网格大小，基于 KITTI 数据集做实验，分别设置不同网格大小实现体素滤波，确定适合大小的网格，如图 3.10。

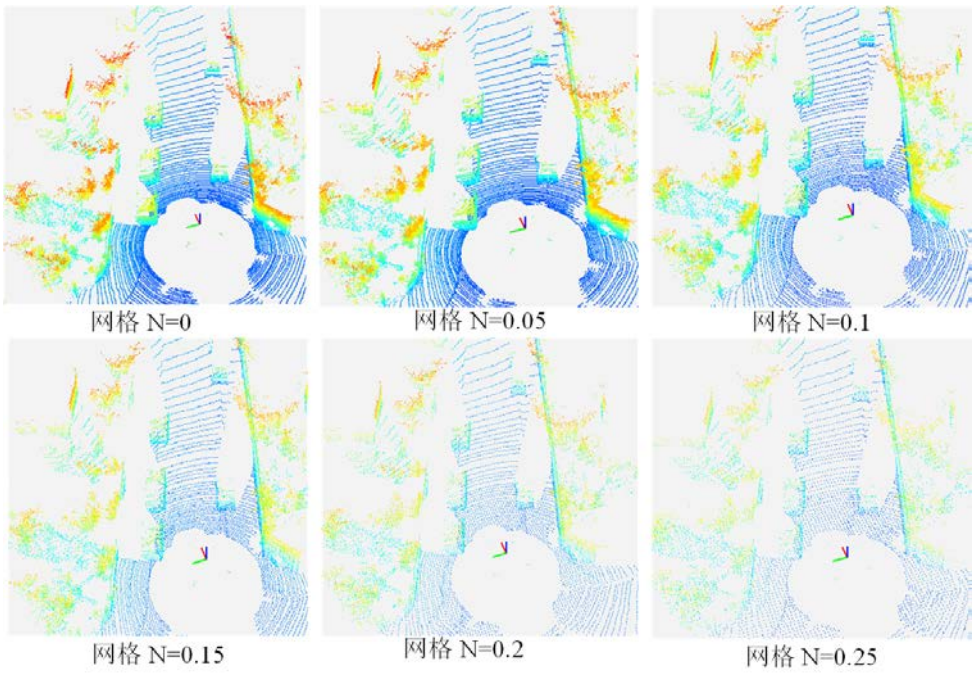


图 3.10 不同滤波参数滤波后点云图

为了更清晰了解体素网格滤波后点云数据的变化，将数据整理成表格，如下表 3.2 所示。

表 3.2 不同滤波参数后点云数目表

网格大小	0.00	0.05	0.10	0.15	0.20	0.25
滤波后点云数	122556	856615	54733	27743	21422	16975

根据表 3.2 和图 3.10 可以看出，当设置体素网格为 0.05 时，从点云图上看变化不大。当增大网格大小为 0.10 时，点云的密度仍然足够高，以至于肉眼无法察觉到明显的稀疏化。当把点云滤波网格设置成 0.15，此刻可以明显看出点云已经变稀疏了，点云中的部分细节开始被丢失，但整体上仍然保留了足够的信息继续增大网格，当网格大小到 0.20 的时候，已经对点云图造成了损失，最后设置网格大小为 0.25 可以明显看出点云损失比较严重。综合考虑，设置滤波的网格大小为 0.15。

3.5.3 实验结果分析与对比实验

（1）算法改进前后平均精度对比

为了验证本文改进算法的性能，基于 KITTI 数据集上与改进前进行对比实验，然后将二者在简单、中等、复杂三种场景下的性能差异以柱形图的形式表现出来。从图 3.11 中可以看出改进后的算法，在各类样本检测上精度均有提升。在行人的检测上，简单场景精度提升 6.22%，中等场景提升 8.02%，复杂场景提升 4.72%。在汽车的三种不同检测场景中，其中简单场景下精度提升 3.21%，中等场景下提升 4.23%，复杂场景下提升 2.53%。对骑行者而言，简单场景提升 4.33%，中等场景提升 6.26%，复杂场景下提升 3.49%。从整体上来看对行人和骑行者较小物体的优化效果大于汽车。猜想是车辆的点云样本数据较多，点云相对密集，特征信息的增强对其性能提升比较有限。对于行人和骑行者这样的小尺寸目标，数据样本和点云密集度都不高，通过增强特征信息使得小目标更加容易被检测到，对小物体检测的性能提升较大。

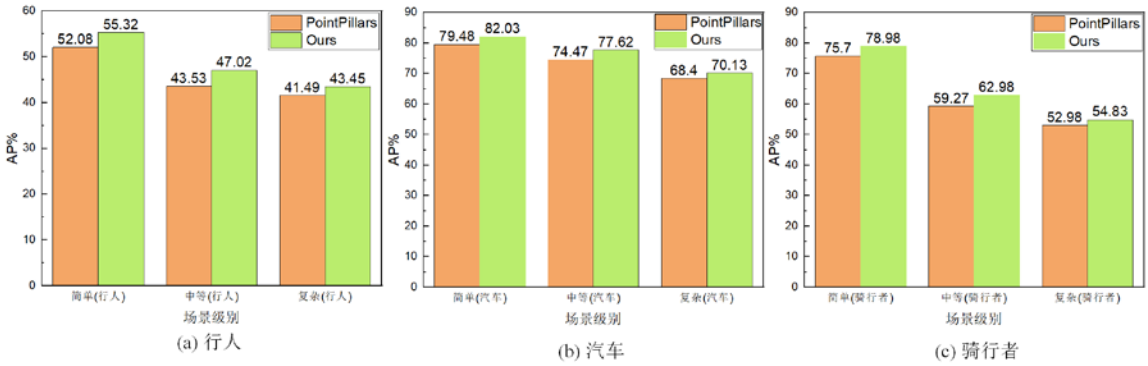


图 3.11 算法改进前后各类平均精度变化

（2）可视化实验

为了进一步深入了解算法改进前后的具体性能差异，分别在多目标场景、远距离场景、长距离场景三种不同场景下进行实验。对实验结果进行可视化展示。在对比实验中绿色 3D 检测框表示汽车检测框，蓝色 3D 检测框表示行人检测框，而黄色框 3D 检测框表示骑行者检测框，对于检测错误的地方使用红色进行了标识。

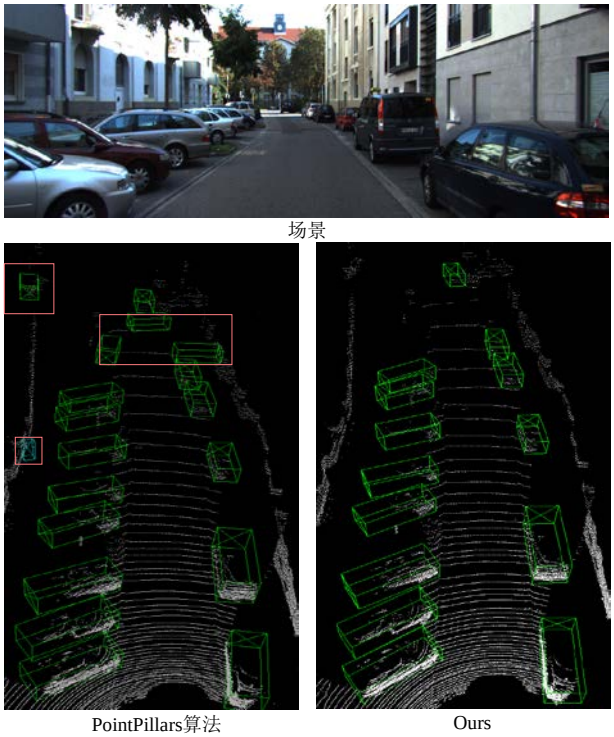
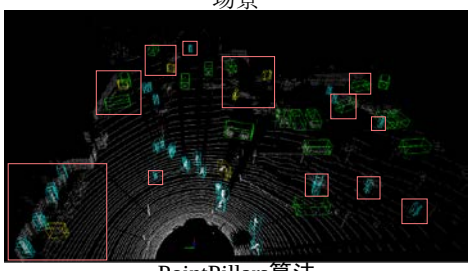
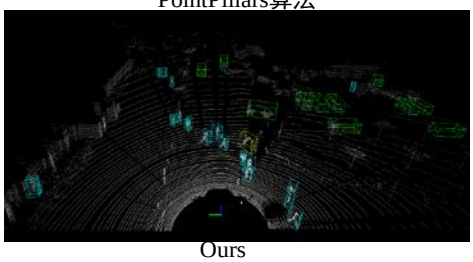


图 3.12 长距离场景检测对比

场景



PointPillars算法

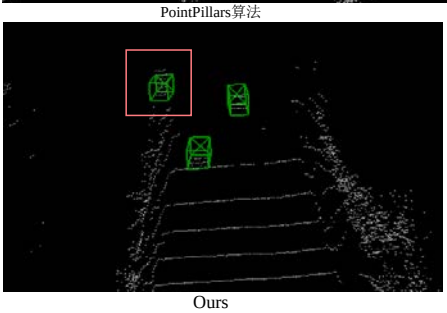


Ours

图 3.13 多目标场景下的检测对比

场景

PointPillars算法



Ours

图 3.14 远距离场景检测对比

图 3.12 是长距离场景 PointPillars 算法对于近处的车辆识别效果较好，但是后半段识别精度较低。在后半段将屋边凸起识别为行人，在远处长有枝条的大树识别为汽车。从图 3.13 可以看出 PointPillars 算法在复杂的多目标环境下，效果比较差，存在多处漏检、误检现象。算法把附近的树木、矮小的柱子等目标错误的检测成了行人或自行车，其中误检为行人的情况较多，改进后的算法改善了此类现象。在远距离场景中，如图 3.14。图中存在三处点云，两辆汽车和一个桥柱。由于距离较远导致目标点云稀疏，PointPillars 算法无法识别到最远处的车辆，导致漏检错误。改进后的算法增强了小物体的检测，成功识别。

从三个实验的结果来看，我们可以得出结论：引入注意力机制、优化目标朝向损失函数以及应用 SoftPlus 激活函数，这三种策略共同实施，显著提高了小目标的识别效果。但是本文算法也存在缺陷，在图 3.14 中改进后的算法将桥柱错误识别为车辆，这表明算法在区分具有相似特征的不同类别物体时，尤其是当这些物体距离较远、细节信息模糊时，其性能还有待提升。后文将结合视觉信息综合判断。

（3）与主流算法对比

通过以上数据可以看出改进后的 PointPillars 算法在 3D 检测任务上相较于原始的 PointPillars 算法在 mAP 上有所提升。为了进一步评估基于 PointPillars 改进算法在点云目标检测任务中的性能表现采用 KITTI 数据集作为基准测试平台，并与当前主流算法进行了对比分析，将数据统计到表格中。从表 3.3、3.4 和 3.5 中各项数据可以看出，基于 PointPillars 改进后的算法在汽车、行人、骑行者三种类别上的检测精度均有提升。其中对于行人和骑行者这种较小的物体提升比较大。与主流的检测算法对比发现改进后算法有很大的优势。

表 3.3 不同算法在车辆检测中的性能比较

方法	Car(IOU=0.7)			
	简单	中等	困难	mAP
AVOD-FPN	81.64	71.85	64.19	72.56
F-PointNets	81.42	70.43	62.78	71.54
VoxelNet	77.47	65.11	57.73	66.77
SECOND	83.24	72.66	65.60	73.83
PointPillars	79.48	74.47	68.40	74.12
Ours	82.03	76.02	69.93	75.99

表 3.4 不同算法在行人检测中的性能比较

方法	Pedestrian(IOU=0.5)			
	简单	中等	困难	mAP
AVOD-FPN	50.8	42.83	40.96	44.86
F-PointNets	51.92	45.03	72.01	56.32
VoxelNet	39.48	33.69	31.50	34.89
SECOND	50.07	42.36	36.29	42.91
PointPillars	52.08	43.53	41.49	45.70
Ours	55.32	47.02	43.45	48.60

表 3.5 不同算法在骑行者检测中的性能比较

方法	Cyclist(IOU=0.5)			
	简单	中等	困难	mAP
AVOD-FPN	64.30	52.56	47.01	54.62
F-PointNets	72.03	56.48	51.23	59.91
VoxelNet	61.62	48.26	44.37	51.42
SECOND	71.51	53.85	46.75	57.37
PointPillars	75.32	60.01	52.31	62.55
Ours	77.27	62.12	54.32	64.57

3.6 本章小结

针对 PointPillars 点云算法对小物体精度低的问题，提出一种基于改进 PointPillars 的激光点云检测算法。在二维卷积特征提取网络中引入了 ECA 注意力机制增强对重要信息的提取能力。由于 ReLU 激活函数无法对负数输入进行处理，会造成一定的有效数据损失，为此引入 Softplus 激活函数提高对数据的处理能力。除此之外，优化目标朝向损失函数提高了对目标朝向的预测精度，提高模型的精确度和鲁棒性。最后，通过平移、镜像、翻转等方法来扩大数据集，以学习到更加丰富的特征信息。同时采用体素滤波来减少点云数据冗余提高计算效率。在 KITTI 公共数据进行测试，mAP 提升了 4.57%。

第4章 基于激光雷达与相机融合的检测与跟踪

4.1 引言

在第 2、3 章中，研究了激光雷达对 3D 点云的目标检测以及相机对 2D 图像的检测，然而从前两章的试验可以看出，基于视觉的检测算法容易受到遮挡、光照等不利环境的影响，稳定性较差。激光雷达虽然能获取准确的 3D 位置信息，但无法提供准确的类别色彩等信息。本章针对轻量化平台算力低的特点，构建一种低复杂度的决策级融合算法。通过该算法完成点云与图像的信息互补，提高检测的精度。检测到目标后通过 DeepSORT 算法进行目标跟踪。

4.2 激光雷达与相机融合检测与跟踪算法结构

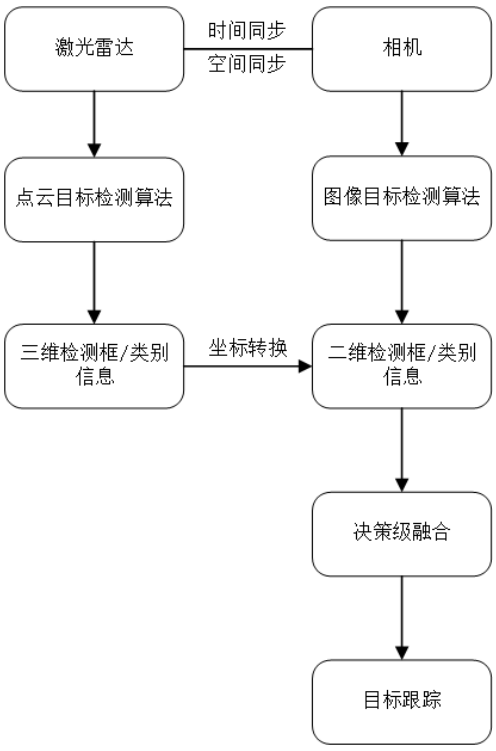


图 4.1 激光雷达相机融合检测与跟踪算法流程

激光雷达与相机融合的检测与跟踪系统整体分成检测、融合、目标跟踪三个部分。检测模块有两个检测器，一个负责检测激光点云和一个负责对图像进行识别。激光点云的检测算法使用改进后的 PointPillars 点云目标检测算法，图像的检测使用 YOLOv5-SG 算法。融合模块从检测模块获得的 3D 点云检测框和 2D 图像检测框，使用激光雷达与相机标定获得的外参矩阵将激光点云检测框投影到图像中，然后使用数据关联算法完成决策级融合。目标跟踪模块是最后一个模块，在前一个融合模块中，检测到了目标后，使用 DeepSORT 算法对目标进行跟踪。

4.3 激光雷达与相机融合

4.3.1 多传感器融合策略

传统感知主要依赖于摄像头、激光雷达等单一传感器来感知周围环境信息，然而不同的传感器都有各自的缺点与局限。比如，高清摄像头能够捕捉丰富的视觉信息，对于路障、行人以及车辆等目标具有显著优势。然而，其性能易受到光照条件、天气变化以及遮挡物的影响。激光雷达以其高精度的三维点云数据在距离测量和目标定位方面表现出色，但对于目标类别的识别效果不佳。为了解决这一问题，越来越多人转向多传感器融合方向。这种技术能有效地将来自不同传感器的信息整合加以互补，降低不确定性，以获得被感知对象一致性的描述，从而形成更为全面的环境认知。通过融合多个传感器的数据，机器人可以更准确、全面地理解周围环境，提高其感知和决策的能力，从而更好地应对各种复杂任务。按照信息处理的程度，可以分成三类，分别是数据级融合、特征级融合、决策级融合三个级别。

数据级融合作为多传感器信息融合的基础层次，旨在将不同传感器采集的原始数据进行直接整合，目前主流的方法有加权平均法[63]、可变权重法[64]等。相较于数据级融合，特征级融合则更注重于从原始数据中提取关键特征，并进行下一步的融合处理，常用的方法有主成分分析法[65]、独立成分分析法[66]、小波变换[67]、神经网络等。决策级融合是将来自不同传感器的数据进行决策级策略处理和集成，主要方法有 D-S 证据推理法[68]、卡尔曼滤波算法等。

（1）数据级融合

数据级融合，亦称像素级融合，属于数据底层的直接整合过程。数据级融合将多个传感器的原始数据直接融合，以形成更为全面和详尽的数据集合。随后，从这些融合数据中提取所需特征，以供后续分析与应用。需要注意的是，进行传感器融合的前提是各传感器观测的是同一物理量，否则需进行校准以确保数据的可比性和准确性。相较于其他融合方式减少了信息融合过程中的信息丢失问题，但在某些方面也带来了新的问题，由于融合后的数据信息量过大会增加信息处理的难度导致计算负载加大。

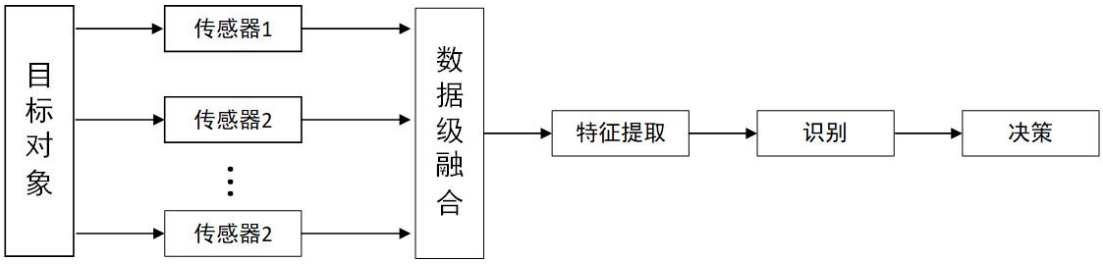


图 4.2 数据级融合流程图

（2）特征级融合

特征级融合属于是中间级的融合，融合过程是传感器先独立提取信息特征，然后将来自多个传感器之间的特征信息通过算法融合在一起形成新的特性，最后将新的特征进行检测得到结果。其中关键的一步是选择合适的特征进行融合，常见的特征有边缘、方向、速度等。特征融合的发展较为完善，往下可以划分成为两类，目标状态融合和目标特征融合。这种融合方式减少了一定的计算量加快了系统的运行速度，但相对的也会带来一定程度的信息丢失问题。

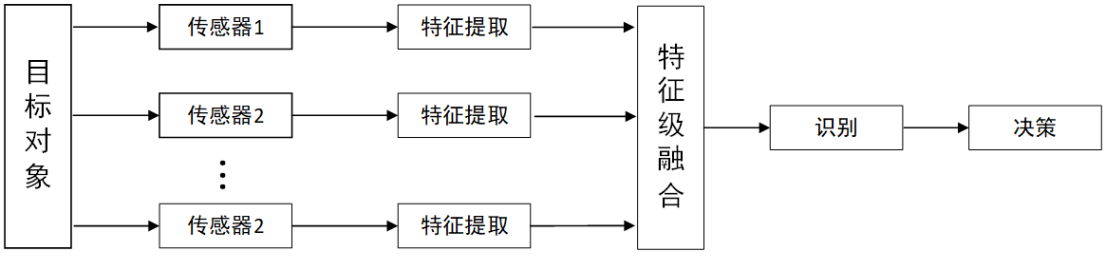


图 4.3 特征级融合流程图

（3）决策级融合

决策级融合是每个传感器对自身提取到的特征信息进行分析，得到结果，然后将所有的结果通过一个算法来进行综合评估来获得一个更为准确的结果。决

策级融合中传感器之间相对独立，若是单个传感器检发生了故障不会影响其他的传感器的检测结果，这种方式对系统的鲁棒性和运行速度都有极大的提升。在硬件层面实现多传感器融合并非难事，真正的挑战在于算法设计。算法是决策级融合的核心，其决定了如何有效地整合不同传感器的信息，以及如何从中提取出最有价值的结果。

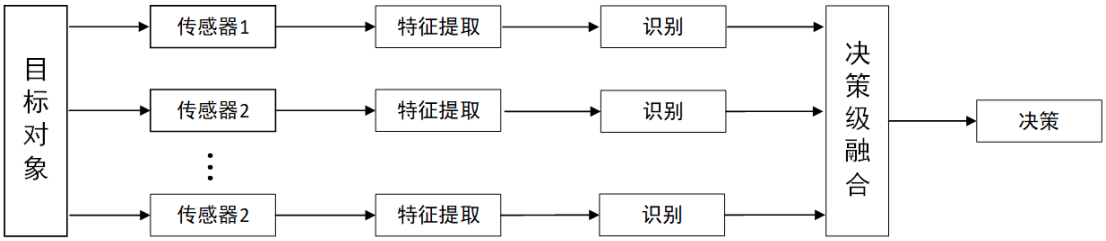


图 4.4 决策级融合流程图

综合比较三种融合方式，本文选择决策级融合的方案。决策级融合对硬件要求相较于其他两种方案要求低，适合嵌入式设备的部署，决策级融合的困难在于算法，是一个很有前景的方向。融合之前需要将检测后的结果框进行时空对齐统一到一个坐标系下，然后使用数据关联算法即可完成融合。

4.3.2 激光雷达相机时空对齐

由于激光雷达与相机的位姿是不同的，故而通过激光雷达与相机获得采集的数据，并不能直接进行融合，在融合之前需要先完成传感器之间的空间对齐与时间对齐。空间对齐是因为激光雷达与相机二者是处于不同坐标系下，为了进行多源数据融合，需要将二者转换到同一坐标系下。相机中有三种坐标系，分别是相机坐标系、图像坐标系、像素坐标系。来自外界的激光点云数据先是转换到相机坐标系下，然后从相机坐标系下转到图像坐标系下，最后转到像素坐标系下。时间对齐是因为各个传感器都有独立的封装并按照各自的时钟基准运行，激光雷达与相机采样频率一般不同，这会导致二者采集的数据在时间上不同步[69]。

（1）空间对齐

相机图像是在以相机光心为中心的坐标系下，而激光点云数据是在以激光雷达为中心的坐标轴下。坐标系的作用都是用来描述位置，差别在与不同坐标系的原点不同，想要实现相机坐标系与激光雷达坐标系的变化只需简单的平移和旋转就可以实现[70]。

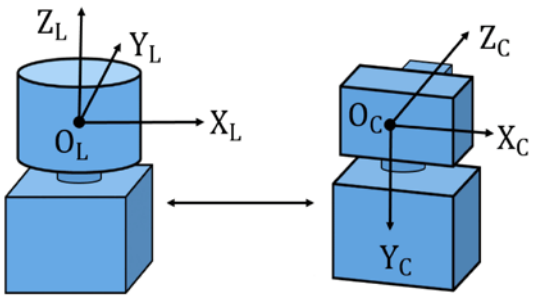


图 4.5 激光雷达与相机传感器坐标转换

如图 4.5 所示，激光雷达的坐标系是$O_L X_L Y_L Z_L$，相机坐标系系是$O_C X_C Y_C Z_C$，要实现两个坐标系的重合需要经过旋转矩阵 R 和平移矩阵 T。

旋转的目的是为了让，两个坐标系的开口方向一致，旋转分成三种分别是绕 X 轴旋转、绕 Y 轴旋转和绕 Z 轴旋转。如下图所示是激光雷达坐标系绕 Z 轴旋转θ度。

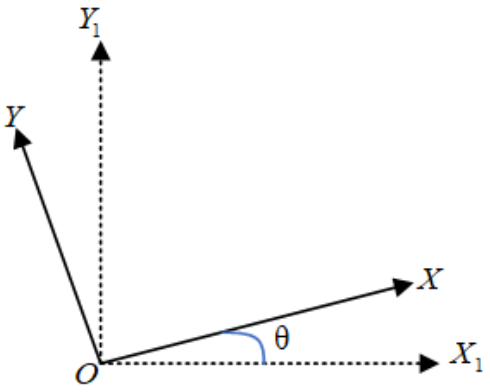


图 4.6 绕 Z 轴旋转

根据二者变化关系可得式 4.1。

$$\begin{cases} X_1 = X\cos\theta - Y\sin\theta \\ Y_1 = X\sin\theta - Y\cos\theta \end{cases} \tag{4.1}$$

整理可得式 4.2。

$$\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ X \\ Z \end{bmatrix} \tag{4.2}$$

设R_1表示绕 Z 轴旋转θ角度的矩阵，R_1的计算公式如式 4.3 所示。

$$R_1=\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \tag{4.3}$$

同理可以得到绕 X 旋转角度为φ旋转矩阵R_2、Y 轴旋转角度为ω旋转矩阵R_3的表达式如下式 4.4 和 4.5 所示。

$$R_2=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\varphi & -\sin\varphi\\ 0 & \sin\varphi & \cos\varphi\end{bmatrix} \tag{4.4}$$

$$R_3=\begin{bmatrix}\cos\omega & 0 & \sin\omega\\ 0 & 1 & 0\\ -\sin\omega & 0 & \cos\omega\end{bmatrix} \tag{4.5}$$

将三个旋转矩阵相乘，得到最终的旋转矩阵R。

经过上面旋转矩阵的变换，激光雷达与相机坐标轴的朝向已经一样的，只需向 X、Y、Z 方向平移Δ单位就可以实现激光雷达坐标系一点与相机坐标系重合，综上所述，激光点云与相机坐标系点转换关系如式 4.6。

$$\begin{bmatrix}X_C\\ Y_C\\ Z_C\end{bmatrix}=R\begin{bmatrix}X_L\\ Y_L\\ Z_L\end{bmatrix}+T \tag{4.6}$$

相机坐标系点$P(X_C,Y_C,Z_C)$通过光心映射到图像坐标系下生成点$\mathrm{p}(x,y)$。

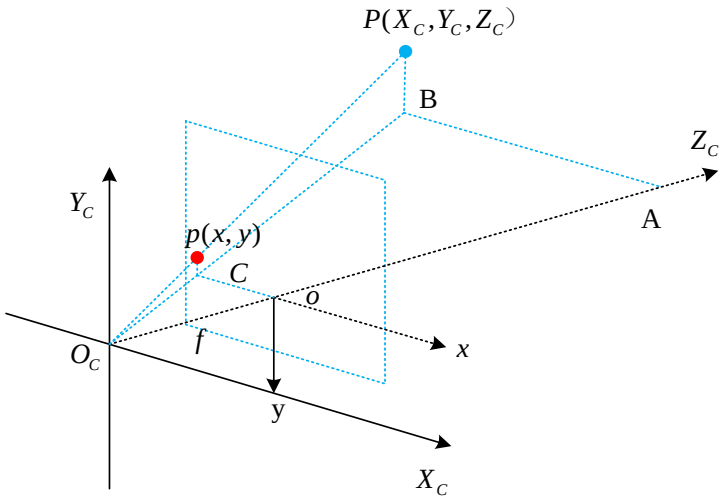


图 4.7 相机坐标系与图像坐标系转换

根据三角形相似原理可以知道 $\Delta ABO_C \sim \Delta oCO_C$、$\Delta PBO_C \sim \Delta pCO_C$ 则可得式 4.7。

$$\frac{AB}{oC}=\frac{AO_C}{oO_C}=\frac{PB}{pC}=\frac{X_C}{x}=\frac{Z_C}{f}=\frac{Y_C}{y} \tag{4.7}$$

根据上述关系列出表达式，如式 4.8 所示。

$$\begin{cases} x=f\dfrac{X_C}{Z_C} \\ y=f\dfrac{Y_C}{Z_C} \end{cases} \tag{4.8}$$

转为为矩阵，结果如式 4.9。

$$Z_C\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}=\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} \tag{4.9}$$

图像坐标与像素坐标都在成像平面中，但是二者的原点与度量单位有差异。图像坐标系的单位是 mm，像素的横坐标 u 与纵坐标 v 分别是在其图像数组中所在的列数与所在行数，二者之间的转换关系是 dx 和 dy 代表了每一列和每一行代表多少 mm。

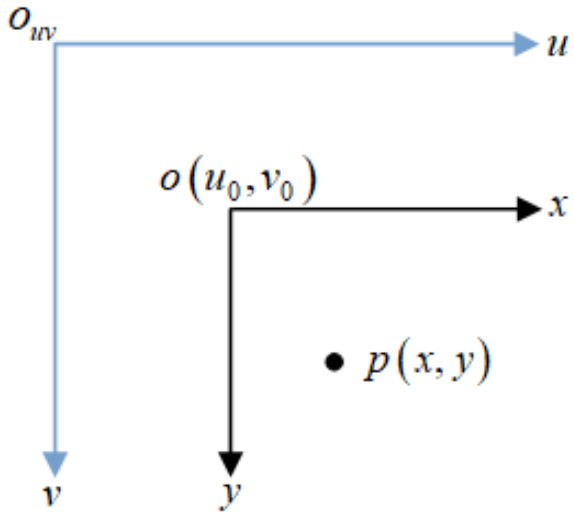


图 4.8 图像坐标系到像素坐标系的转换示意图

根据二者的转换关系可得式 4.10。

$$\begin{cases} u = \dfrac{x}{dx} + u_0 \\ v = \dfrac{y}{dy} + v_0 \end{cases} \tag{4.10}$$

整理可得式 4.11。

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \dfrac{1}{dx} & 0 & u_0 \\ 0 & \dfrac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4.11}$$

综上所述得到激光雷达转像素坐标关系如式 4.12。

$$Z_C \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} \tag{4.12}$$

$$K = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.13}$$

$$T = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \tag{4.14}$$

K 表示为内参矩阵，T 表示为外参矩阵。

（2）时间对齐

时间对齐是指将多个数据源中的时间信息同步或校准，以确保在相同的时间轴上对齐。在系统运行的时候，激光雷达和相机通常是独立运行的系统，二者数据获取频率大概率不同，有的甚至有很严重的延迟。为此，在进行数据融合之前，需要对二者的时间信息进行校准，以保证二者在同一时刻收集的数据是同一个。目前，时间同步有两种方法，一是硬同步这是添加一个外部触发器，这样二者就可以在同一时间进行数据采集。二是软件同步，通过算法将收集好的数据再进行重新校准。

4.3.3 数据关联

在激光雷达与相机的时间与空间对齐之后，建立起二者的转换关系，这样图像和点云共享同一坐标系且时间上保持数据的一致性。数据关联是点云与图像信息融合中一个关键方面，关联函数对于数据关联来说起着至关重要的作用。在当前使用比较多的是基于 IOU 交并集方法，主要关注点在与对象的几何信息。点云的三维包络框使用上文联合标定得到的参数投影到图像平面形成二维的包络框，通过比较图像检测框与点云投影框之间的关系完成目标的匹配。

投影包络框的坐标为(x_l, y_l, x_l', y_l')，图像检测框的坐标为(x_c, y_c, x_c', y_c')。

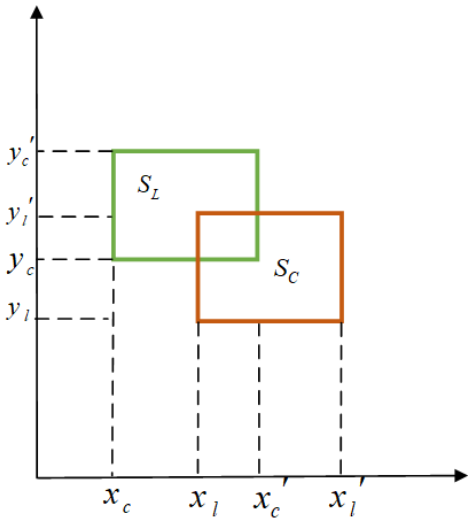


图 4.9 投影包络框与图像检测框位置示意图

点云投影框S_L和图像检测框的面积S_C如式 4.15 和 4.16 所示。

$$S_L = (x_l' - x_l) \times (y_l' - y_l) \tag{4.15}$$

$$S_C = (x_c' - x_c) \times (y_c' - y_c) \tag{4.16}$$

投影包络框和图像检测框交面比 *IOU* 计算公式如式 4.17。

$$IOU = \frac{S_L \cap S_C}{S_L \cup S_C} \tag{4.17}$$

$0.5 \le IOU \le 1$，表明两个目标检测框来自于同一个目标，则将相机检测的类别和位置信息作为车辆的输出信息。

$0 < IOU < 0.5$，表明虽然真实目标存在，但是由于实际环境中场景的复杂多

变性，可能出现晴天、雨天、雪天、大雾等不同的天气状况，此时需要考虑相机受天气的影响可能会存在漏检的情况，最终将相机的检测类别和雷达检测的方位信息作为车辆输出信息。

$IOU=0$ 时，表明两个目标检测框不存在交叉，需要进一步讨论。$S_C \neq 0$且$S_L = 0$，表明雷达漏检，最终输出相机信息。$S_L \neq 0$且$S_C = 0$，表明相机漏检，输出雷达检测信息。$S_L = 0$且$S_C = 0$，表明目标不存在，判断结束。

4.4 多目标跟踪

在目标检测过程中，复杂的道路环境以及目标姿态的持续变化等因素，常常导致目标在连续帧中遭遇漏检，进而引发目标的短暂丢失。为确保智能系统稳定，引入目标跟踪算法来弥补目标检测的不足显得尤为重要。目标跟踪算法可以利用目标在连续帧间的运动信息和特征关联，对目标进行持续跟踪，并在目标被漏检时，基于先前的轨迹和状态信息对目标位置进行估计和预测。不同于 DeepSORT 算法使用 Faster R-CNN 来获取目标对象，本节利用激光雷达相机融合算法作为检测器，来实现多目标跟踪。

DeepSORT 算法是目标跟踪领域中常用的算法，是从 SORT 算法基础上发展而来，与之不同的是其增加了级联匹配和新轨迹的确认。DeepSORT 算法首先利用卡尔曼滤波器对目标的轨迹进行预测，然后将这些预测得到的轨迹与当前帧中通过目标检测算法得到的检测目标进行匹配。匹配过程包括级联匹配和 IOU 匹配，级联匹配使用外观模型和运动模型计算相似度，得到代价矩阵，并通过门控矩阵限制代价矩阵中过大的值。匹配是一个循环，从没有丢失过的轨迹开始，逐步匹配到丢失时间最长的轨迹。新轨迹确认可以将被遮挡的目标找回，降低被遮挡然后再出现的目标发送 ID Switch 次数。

4.4.1 卡尔曼滤波状态估计

卡尔曼滤波算法是一种状态估计算法，一般应用于从含有噪声和不确定性因素的观测数据中提取真正有效的数据，是一种是相当经典的滤波算法。卡尔曼滤波算法通过上一时的预测值和当前时刻观测值，预测出下一时刻最优估计值。

首先定义状态变量 x_k 和观测变量 z_k 为：

$$x_k = [x, y, w, h, v_x, v_y, v_w, v_h] \tag{4.18}$$

$$z_k = [x, y, w, h]^T \tag{4.19}$$

x 和 y 分别代表其当前帧预测框的中心点坐标的横坐标和纵坐标，w 和 h 则表示宽度和高度，v_x、v_y、v_w、v_h，描述了目标边界框中心点在水平方向的速度、垂直方向的速度，以及当前时刻边界框宽度和高度相对于上一时刻的变化率。

卡尔曼滤波算法的数学状态方程如下：

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{4.20}$$

$$z_k = Hx_k + v_k \tag{4.21}$$

上面公式中，x_k 状态矩阵；A 状态传递矩阵；z_k 是状态转移矩阵观测值；B 控制输入矩阵；w_{k-1} 过程噪声；v_k 高斯白噪声；满足以下条件的时候使用 Q 和 R 代替 w_{k-1} 和 v_k 的协方差。

$$\begin{aligned} p(w) &\in N(0, Q) \\ P(v) &\in N(0, R) \end{aligned} \tag{4.22}$$

卡尔曼滤波分成预测和更新两个步骤，这两个步骤共同构成了卡尔曼滤波算法的递归过程，完成系统动态估计这一任务。预测步骤基于系统的动态模型，对状态进行先验估计，并计算相应的协方差以量化预测的不确定性。更新步骤利用当前的观测数据，通过计算卡尔曼增益来融合预测值与观测值，从而得到后验估计，即更新后的状态估计值及其协方差。这两个步骤交替进行，不断迭代，使得卡尔曼滤波能够在不断变化的系统环境中，实时地、准确地估计系统的状态。

状态预测方程：

$$\hat{x}_{\bar{k}} = A\hat{x}_{k-1} + Bu_{k-1} \tag{4.23}$$

协方差预测方程：

$$P_{\bar{k}} = AP_{k-1}A^T + Q \tag{4.24}$$

式中，$\hat{x}_{\bar{k}}$ 表示当前时刻的先验估计值，A 表示状态转移矩阵，B 表示控制矩阵，u_{k-1} 表示输入控制量，$P_{\bar{k}}$ 为 $\hat{x}_{\bar{k}}$ 的协方差，Q 表示过程噪声的协方差。

状态更新方程如下：

$$K_k = \frac{P_{\bar{k}} H^T}{H P_{\bar{k}} H^T + R} K_k \tag{4.25}$$

$$\hat{x}_k = \hat{x}_{\bar{k}} + K_k(\mathrm{z_k} - H\hat{x}_{\bar{k}}) \tag{4.26}$$

$$P_k = (I - K_k H) P_{\bar{k}} \tag{4.27}$$

K_k 表示卡尔曼增益值，H 表示观测矩阵，R 表示观测噪声的协方差，$\hat{x}_{\bar{k}}$ 表示后验状态估计值，$\mathrm{z_k}$ 为实际观测值，$H\hat{x}_{\bar{k}}$ 为预测值，P_k 为 $\hat{x}_{\bar{k}}$ 的协方差。

4.4.2 数据关联

匈牙利算法是一种图论算法，是一种用来解决二分图最大匹配问题的算法。在 DeepSORT 算法中中负责连续帧目标的关联匹配问题。匈牙利算法通过不断选择关联矩阵中的零值，并确保每行和每列只选择一个零值，以最小化总体的匹配代价。通过迭代的方式，匈牙利算法能够找到一个最优的分配方案，使得总体的匹配代价最小，从而实现目标跟踪的最佳性。DeepSORT 算法在目标跟踪过程中，充分融合了目标的运动信息和外观特征。通过整合这两种互补的信息源，DeepSORT 显著提升了在复杂场景中识别、关联和持续跟踪目标对象的准确性。

（1）运动信息关联

运动信息关联是指通过计算预测状态与检测状态之间的马氏距离，预测状态，以此对目标进行运行信息分析。判断出目标下一时刻出现的位置。

$$\mathrm{d}^{(1)}(i,j) = (d_j - y_i)^T S_i^{-1} (d_j - y_i) \tag{4.28}$$

$$b^{(1)}_{i,j} = \prod[d^{(1)}(i,j) <= t^{(1)}] \tag{4.29}$$

d_j 表示第 j 个目标检测框，y_i 表示第 i 个轨迹，S_i^{-1} 表示 d_j 与 y_i 之间的协方差矩阵。$\mathrm{d}^{(1)}(i,j)$ 与阈值 $t^{(1)}$ 比较，前者小表示匹配成功，否则匹配失败。

（2）外观信息关联

外观信息为 DeepSORT 算法提供了目标的视觉特征，这些特征在目标受到遮挡或场景复杂时能够辅助算法进行目标的匹配和跟踪。DeepSORT 利用卷积神经网络作为特征提取网络，从目标的图像数据中提取出判别性强的特征表示。在计算外观之间的关联性时，DeepSORT 采用余弦距离作为度量指标。

$$d^{(2)}(i,j)=\min\{1-r_j^T r_k^{(i)} \mid r_k^{(i)} \in R_i\} \tag{4.30}$$

式中，r_j 表示第 j 个检测结果的特征向量，r_k^j 表示成功跟踪 k 次的特征集，$r_j^T r_k^{(i)}$ 表示两向量的余弦相似度。

余弦距离也通过特定的阈值来判断是否信息匹配成功，当余弦距离小于阈值 $t^{(2)}$ 表示匹配成功，即当前帧的目标与跟踪列表中的某个已有目标在外观上是相似的，判别公式如下 4.31 所示：

$$b_{i,j}^{(2)}=\prod[d^{(2)}(i,j)\le t^{(2)}] \tag{4.31}$$

通过马氏距离与余弦距离线性加权，当成检测框与跟踪框之间的衡量指标计算公式如下 4.32 所示：

$$c_{i,j}=\lambda d^{(1)}(i,j)+(1-\lambda)d^{(2)}(i,j) \tag{4.32}$$

其中，$d^{(1)}(i,j)$ 是马氏距离，$d^{(2)}(i,j)$ 是余弦距离，λ 是控制两种距离度量方式的影响因子。

DeepSORT 通过结合运动信息和外观信息，并使用级联匹配策略，实现了对目标的准确、鲁棒跟踪。级联匹配将经过卡尔曼滤波得到的预测轨迹与检测框进行匹配，匹配的结果有：未匹配轨迹、未匹配检测和匹配成功。

匹配成功的使用卡尔曼滤波更新，然后进行下一帧的预测。对于未匹配轨迹和未匹配的检测，使用 IOU 进行第二次匹配，匹配结果也分三种：未匹配轨迹、未匹配检测输出和匹配轨迹。匹配轨迹的话就同上面的一样使用卡尔曼滤波进行更新和预测，未匹配的轨迹要是没有超过设定 max_age 就进行更新预测，超过了就删除轨迹。对于未匹配的检测框将之认为检测出了一个新目标，开辟出一个新物体轨迹然后对其进行更新和预测。

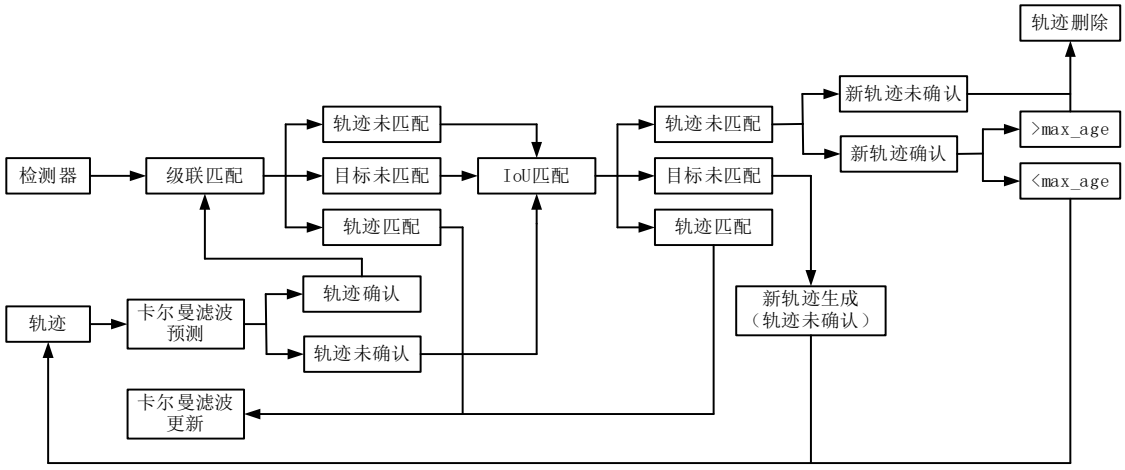


图 4.10 DeepSORT 算法流程图

4.5 实验验证与分析

4.5.1 激光雷达相机融合算法验证平台

本节将激光雷达与相机融合算法进行实验验证与分析，本实验软件算法的框架主要基于 ROS 搭建，对激光雷达与相机的检测任务进行了封装，在调用时发布到各自的处理单元。硬件方面 CPU 选择 Intel Xeon E5-2680 v4，显卡选择 NVIDIA GTX 2080Ti。实验验证的数据集选择 KITTI 数据集。

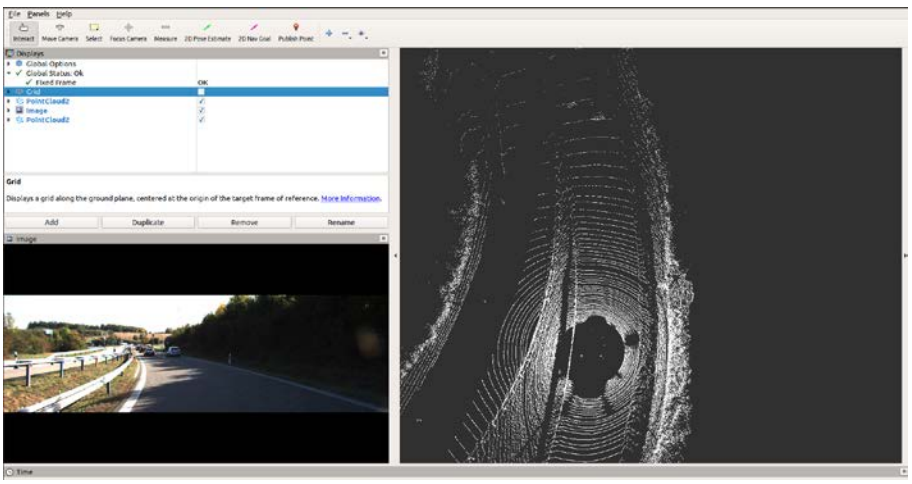


图 4.11 融合算法验证平台

4.5.2 激光雷达相机融合识别实验

完成相机与雷达的联合标定之后，就可以把每帧相对应的激光雷达投影到对应的图像上，实现激光雷达与相机传感器的数据融合。下面从 KITTI 数据集选择三组数据，测试如下图所示。

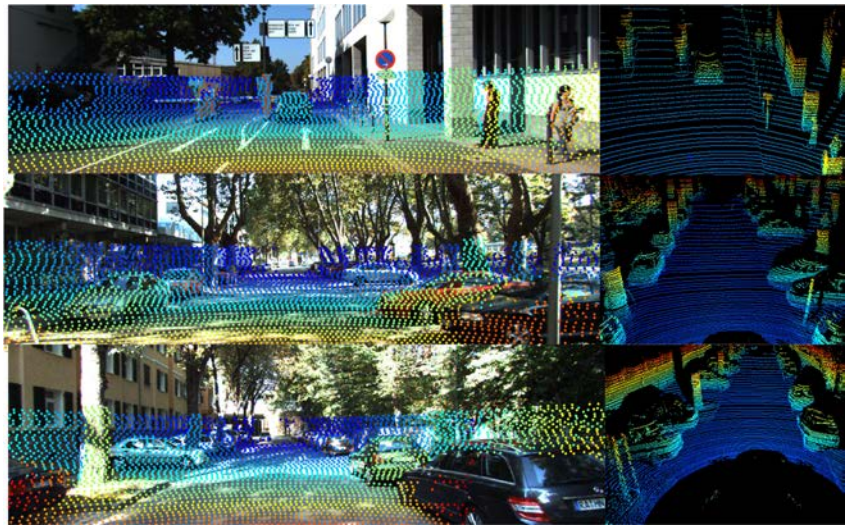
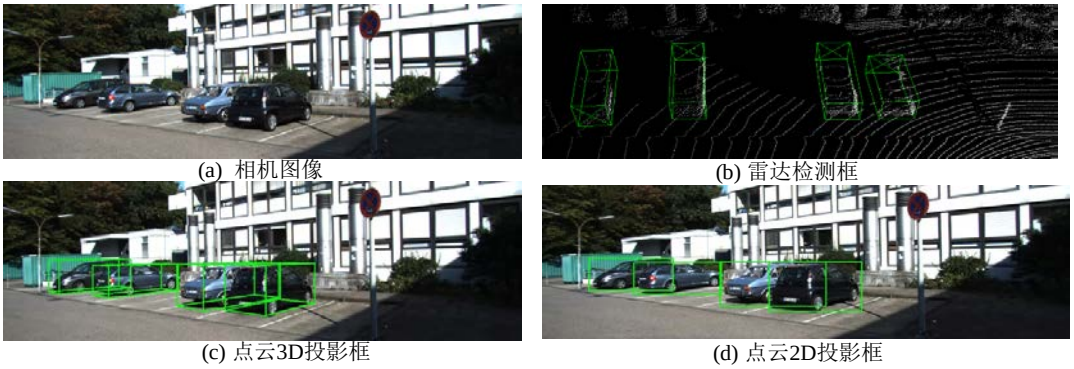


图 4.12 KITTI 数据集投影结果

从图 4.12 可以知道，激光雷达和相机联合标定后，点云和像素之间可以对应起来，有很高的对齐度。扫描到的树、车等点云经过坐标转换关系后，可以与图中的像素点对齐，实现了两个传感器在同一坐标系下对同一目标进行扫描。

将经过 PointPillars 算法得到的激光点云 3D 检测框投影到图像上，投影过程如 4.13。4.13(b)使用联合标定的参数可以将点云三维检测框投影到图像上形成 4.13(c)，然后使用坐标变化关系可以得到 4.13(d)点云 2D 投影框。



(a) 相机图像　(b) 雷达检测框

(c) 点云3D投影框　(d) 点云2D投影框

图 4.13 3D 点云检测框投影过程

为了对检测结果有一个直观的判断，对场景数据在视觉、点云和融合检测下的目标检测结果分别进行可视化，有助于分析系统目标检测的有效性，以及仍存在的缺陷等。图像使用 YOLOV5-SG 算法进行检测，检测到的到的目标使用红色框表示。在点云检测中使用改进的 PointPillars 算法生成 3D 检测框，然后将 3D 检测框投影到图像平面，投影结果使用黄色检测框代表。在融合结果中，最终的结果使用红色检测框代表。

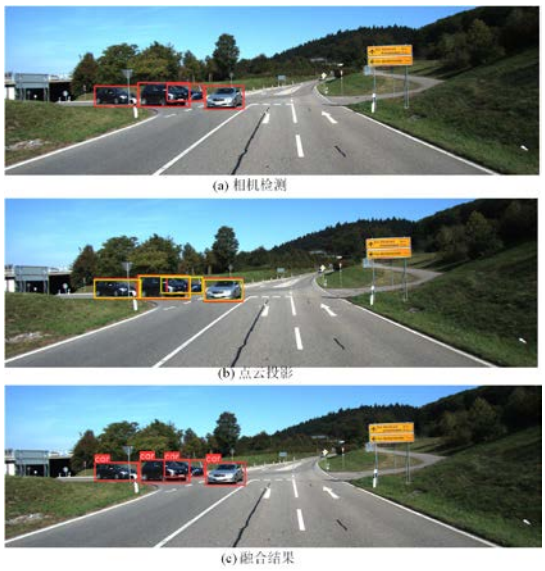


图 4.14 测试场景一

图 4.15 测试场景二

图 4.14 中测试场景一中存在四辆汽车，使用视觉检测算法，检测到了四个车辆目标，3D 点云检测到了四个物体，投影至图像上生成四个黄色投影检测框，此时图像检测框与点云投影框之间的交面比大于阈值，根据设定的决策级融合算法会将相机的检测信息输出。图 4.15 中场景二中由于受到了光照条件的影响，单一的视觉检测性能受到了影响，存在着漏检行为，点云检测不受光照条件的影响，检测到了所有的物体。图 4.15 测试场景二中右侧三个车辆图像与点云检测框 IOU 大于阈值，输出三个相机检测结果，另外一个只存在点云检测框会输出点云检测结果，最终的融合结果是检测出 4 辆汽车。由此可知本文的融合算法在丢失单一传感器数据时仍能够正常完成障碍物检测任务，防止因传感器非正常工作而严重影响检测结果，保证了感知系统在特定条件下的可靠性，提升了整体的容错率。

为了深入研究相机传感器、激光雷达、融合感知之间的差异性，本节从 KITTI 测试集中选取 500 帧图像和点云数据，一共有 1606 个目标将这些分别进行相机检测、点云检测、融合检测，结果统计如表 4.1 所示：

表 4.1 三种检测方法结果对比

检测方法	正检数	误检数	漏检数	准确率
相机检测	1337	102	167	83%
雷达检测	1350	140	116	84%
融合检测	1438	72	96	89%

使用相机进行视觉检测，准确率在 83%，点目标检测正确率在 84%，融合点云与图像信息进行检测的准确率在 89%。说明了相比于使用单一传感器进行目标检测，基于激光雷达与相机信息融合的目标检测算法在准确率上更胜一筹，这在实际应用中具有重要的价值和意义。

4.5.3 激光雷达相机融合目标跟踪实验

（1）目标跟踪评价指标

在本节中，选用了 MOT Benchmark 的三个核心评价指标，以衡量多目标跟踪系统的效能。这三个关键指标包括：多目标跟踪准确率(MOTA)、多目标跟踪精确率(MOTP)以及 ID 切换次数(IDS)。

跟踪准确度主要衡量位置误差，在设计评价函数的时候考虑了多目标跟踪的三种错误，MOTA 的值越大表明该算法的跟踪准确率越高。

$$MOTA = 1 - \frac{\sum FP + \sum FN + \sum IDS}{\sum T} \tag{4.33}$$

FN 是漏检，*FP* 是虚检，*IDS* 是障碍物目标 ID 变化次数，*T* 是真实障碍物数量。

多目标跟踪精确度指的是跟踪过程中真实框与预测框的误差，如下公式所示。

$$MOTP = \frac{\sum_i d_i}{\sum c} \tag{4.34}$$

$\sum c$ 表示所有帧匹配对个数，$\sum_i d_i$ 是所有匹配对中的匹配误差。

IDS 代表 ID Switch，即目标 ID 切换的次数，IDS 越小表示跟踪的稳定性能越好。

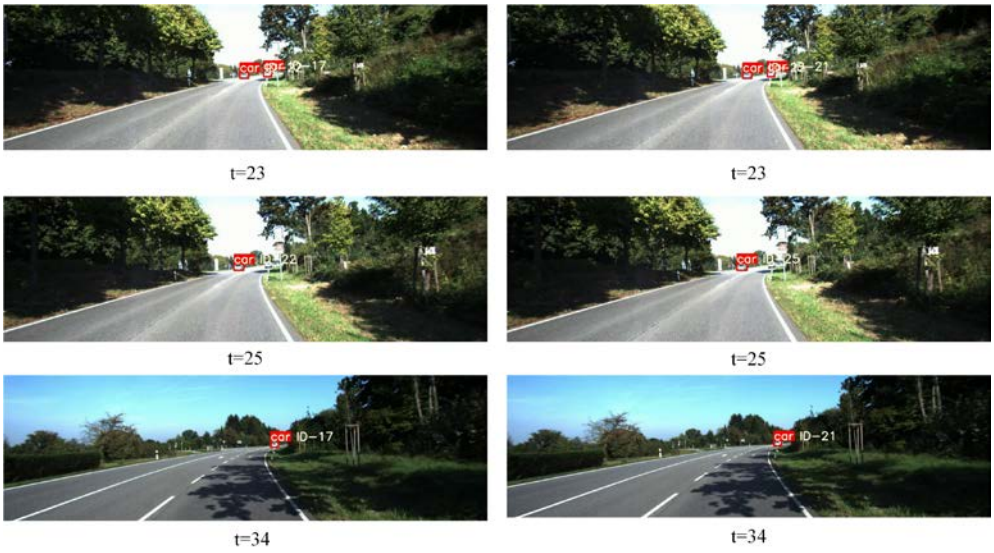
（2）性能分析

为了验证本章算法在 KITTI 数据集下的效果，如图 4.16。从 KITTI 数据集中一个车流量大的场景选取了不连续的 6 个时间点的跟踪结果，将跟踪结果展示如下。以 id 为 55 的车位观察对象在 t 为 12、18、23 这三个时刻中没有出现遮挡等现象，目标跟踪性能良好。当 t 为 26 这个时间点的开始出现遮挡，t 为 28 的时候与 id 为 53 汽车重叠在一起但是依旧保持着轨迹跟踪。等到 t 为 30 的时候，完全被车遮挡住了，这个时候依旧可以正常跟踪。信息融合跟踪算法克服了单一传感器的局限性，在遮挡严重的情况下依旧可以稳定发挥，有很强的鲁棒性。

图 4.16 激光雷达与相机融合目标跟踪测试

（3）对比实验

为了更好的探究基于信息融合的 DeepSORT 算法的跟踪性能，本节利用基于相机信息的 DeepSORT 算法在同样的场景下进行对比实验，相机使用 YOLOv5-SG 算法。



(a) 激光雷达与相机融合的目标跟踪　　(b) 基于相机的目标跟踪

图 4.17 简单场景下目标跟踪性能对比

(a) 激光雷达与相机融合的目标跟踪　　(b) 基于相机的目标跟踪

图 4.18 复杂场景下目标跟踪性能对比

从图 4.17 中有 23、25、34 三个时刻数据，在 t 为 23 的时候检测到了两个物体 ID，等到 t 为 25，有一个 id 为 17 的车辆发生丢失了，最后 t 为 34 的时候丢失的 ID 的目标重新找回。基于相机信息的目标跟踪与基于信息融合的目标跟踪在简单场景中都能发挥很好的效果。从图 4.18 中可以看出，t 为 6 的时刻可以看出两种算法对多种目标识别效果相差无几。t 为 7 的时候，随着检测场景的复杂度提升，二者之间的差异就开始体现出来了，基于信息融合的检测到一个坐在地上的目标 47，基于相机信息的目标跟踪算法则将之遗漏了。最后 t 为 8 的时候，信息融合的目标跟踪算法对 t 为 6 和 t 为 7 两个时刻检测到的所有对象进行跟踪。基于相机的跟踪算法，依旧遗漏了 t 为 7 时刻出现的目标。由于信息融合算法融合了激光点云的检测信息，极大改善了视觉传感器易受光线和遮挡影响的弊端。

（4）同其他主流算法对比

通过对 KITTI 多目标跟踪数据集中的测试序列进行算法评估，并与一系列优秀的多目标跟踪算法进行对比，发现本文提出的跟踪算法在多项关键指标上均展现出了卓越的性能。结果如表 4.2 所示。

表 4.2 KITTI 多目标跟踪测试结果

多目标跟踪算法	MOTA(%)	MOTP(%)	IDS
Complexer-YOLO	75.70%	78.46%	1186
DSM	76.15%	83.42%	296
SASN-MCF nano	70.86%	82.65%	443
SSP	72.72%	78.55%	185
CIWT	75.39%	79.25%	165
MDP	76.59%	82.10%	130
LP-SSVM	77.63%	77.80%	62
FANTrack	77.72%	82.32%	150
NOMT	78.15%	79.46%	31
MCMOT-CPD	78.90%	82.13%	228
JCSTD	80.57%	81.81%	61
Ours	76.80%	83.39%	152

4.6 本章小结

针对复杂环境下单一传感器的目标检测与跟踪受限问题，提出了一种新的

激光雷达与相机信息融合方法。该算法通过设计的决策级融合策略完成点云和视觉目标匹配，融合激光雷达与相机的信息提高了检测精度，减少了光线、遮挡带来的问题。检测到目标后，通过 DeepSORT 算法进行目标跟踪，能够在长期 ID 丢失的情况下，表现出较好的性能。通过在 KITTI 数据集上测试，结果表明提出的算法性能相比于单一传感器，具有很大优势，在遮挡、光线不足之处更好的完成目标检测与跟踪任务。与多种主流目标跟踪算法相比较，结果显示该算法在性能上依然具有显著的优势。

第5章 总结与展望

5.1 总结

本文聚焦于自动导航机器人的环境感知系统，针对户外环境复杂多变、单一传感器在目标检测与跟踪中存在的局限性，设计了一种新的激光雷达与相机融合感知的目标检测与跟踪系统。该系统旨在通过结合激光雷达的高精度空间测量能力和相机的丰富颜色和纹理信息，提升自动导航机器人在复杂环境中的感知性能。本文工作如下：

（1）针对现有基于深度神经网络的视觉目标检测算法高复杂度与嵌入式平台低算力的矛盾，提出一种基于改进 YOLOv5 的的轻量级目标检测算法，首先，使用 Ghost 卷积和 C3-Ghost 模块来替换原来的普通卷积和 C3 模块，减少模型参数和计算量，加快网络的推理速度。然后，引入 SimSPPF 激活函数使得模型更加高效地进行特征提取和池化操作，从而在保持准确性的同时提升了运行速度。此外，在骨干网络中引入无参注意力机制在不增加参数规模的情况下增强模型特征提取能力。在 KITTI 数据集上进行测试，模型参数量下降 46.6%，同时 mAP 达到了 91.6%，比原始模型提升了 2.35%。

（2）针对点云检测算法对小目标存在漏检、误检的问题，提出了一种改进的 PointPillars 激光点云目标检测算法。首先，将用于特征提取的 2D 骨干网络与 ECA 模块相结合，实现伪图像中位置特征信息的增强和背景噪声等不相关特征信息的弱化。其次，引入 Softplus 激活函数，Softplus 激活函数可以对负数的点云数据进行处理，增强了模型对有效信息的处理能力。最后，通过优化目标朝向损失，进一步提升了模型的精度和鲁棒性。再者，激光点云由于数据过于庞大，选择体素滤波对其进行处理，有效降低的数据的冗余度，减少了算力消耗。在 KITTI 数据集上测试 3D 目标检测性能，该算法在 mAP 上提升 4.57%。

（3）针对复杂环境下单一传感器性能受限问题，设计了一种计算量较小的的决策级融合算法，利用视觉与激光雷达的互补性，提高目标检测与跟踪的鲁棒性和精度。检测到目标后，利用 DeepSORT 算法进行目标跟踪。通过 KITTI 公共数据集对激光雷达与相机融合算法进行验证，结果表明同传统单模态目标检

测与跟踪方法相比可对目标的检测与跟踪更加精确，最后基于 KITTI 数据集与多种主流目标跟踪算法相比较，结果显示该算法在性能上依然具有显著的优势。

5.2 展望

本文对基于激光雷达与相机感知融合的目标检测与跟踪技术进行了初步的研究与分析，未来将从以下几点入手。

（1）针对点云数据的目标检测任务，本文初步采用了基于深度学习的方法。然而，由于神经网络结构的复杂性，其对设备的实时性能产生了不利影响。为了优化实时性能并满足实际应用的需求，未来计划探索并应用传统的聚类算法进行点云目标检测，以期在保持检测精度的同时，提高算法的运算速度和实时响应能力。

（2）在融合策略的探索中，初步尝试了决策融合方法，但其在提升融合性能方面仍有局限。为进一步优化融合效果，可以采用数据级别融合策略，直接在数据层面整合优化信息，更精确地提取目标特征，降低误检和漏检概率，提升目标检测的准确性和可靠性，并增强系统的鲁棒性。

（3）本文只融合了激光雷达与相机传感器，获得的环境数据还不够丰富，计划在未来研究中引入毫米波雷达，实现激光雷达、相机与毫米波雷达的多传感器融合，进而提升感知系统的精度和可靠性。

参考文献

[1] Yao P, Sui X, Liu Y, et al. Vision-based environment perception and autonomous obstacle avoidance for unmanned underwater vehicle[J]. Applied Ocean Research, 2023, 134: 103510.

[2] Li J, Qin H, Wang J, et al. Open streetmap-based autonomous navigation for the four wheel-legged robot via 3d-lidar and ccd camera[J]. IEEE Transactions on Industrial Electronics, 2021, 69(3): 2708-2717.

[3] Zhang L, Xu W, Shen C, et al. Vision-based on-road nighttime vehicle detection and tracking using improved HOG features[J]. Sensors, 2024, 24(5): 1590-1599.

[4] Senel N, Kefferpütz K, Doycheva K, et al. Multi-sensor data fusion for real-time multi-object tracking[J]. Processes, 2023, 11(2): 501-513.

[5] Zou Z, Chen K, Shi Z, et al. Object detection in 20 years: A survey[J]. Proceedings of the IEEE, 2023, 111(3): 257-276.

[6] Ouaknine A. Review of deep learning algorithms for object detection[J]. Medium, 2018, 5: 110-118.

[7] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014: 580-587.

[8] He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE, 2015, 37(9): 1904-1916.

[9] Girshick R. Fast R-CNN[C]. International Conference on Computer Vision, IEEE, 2015: 1440-1448.

[10] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks[J]. Advances in Neural Information Processing Systems, 2015, 28.

[11] He K, Gkioxari G, Dollár P, et al. Mask R-CNN[C]. Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2017: 2961-2969.

[12] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016: 779-788.

[13] Liu W, Anguelov D, Erhan D, et al. SSD: Single shot multibox detector[C]. European Conference Proceedings Computer Vision(ECCV), Springer, 2016, 1(14): 21-37.

[14] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, 42(2): 318-327.

[15] Adarsh P, Rathi P, Kumar M. YOLO v3-Tiny: Object Detection and Recognition using one stage improved model[C], 2020 6th International Conference on Advanced Computing and

Communication Systems (ICACCS). IEEE, 2020: 687-694.

[16] Jiang P, Ergu D, Liu F, et al. A Review of Yolo algorithm developments[J]. Procedia Computer Science, 2022, 199: 1066-1073.

[17] Wang C Y, Bochkovskiy A, Liao H Y M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2023: 7464-7475.

[18] Jiang P, Ergu D, Liu F, et al. A review of YOLO algorithm developments[J]. Procedia Computer Science, 2022, 199: 1066-1073.

[19] Xu N, Qin R, Song S. Point cloud registration for LiDAR and photogrammetric data: A critical synthesis and performance analysis on classic and deep learning algorithms[J]. ISPRS Open Journal of Photogrammetry and Remote Sensing, 2023, 8: 100032.

[20] Jin X, Yang H, He X, et al. Robust LiDAR-based vehicle detection for on-road autonomous driving[J]. Remote Sensing, 2023, 15(12): 3160.

[21] Qi C R, Su H, Mo K, et al. Pointnet: Deep learning on point sets for 3D classification and segmentation[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2017: 652-660.

[22] Shi S, Wang X, Li H. Pointrcnn: 3D object proposal generation and detection from point cloud[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2019: 770-779.

[23] Shi S S, Wang Z, Shi J P, et al. From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network[J]. Transactions on Pattern Analysis and Machine Intelligence, IEEE, 2021. 43(8): 2647-2664.

[24] Yang Z, Sun Y, Liu S, et al. 3DSSD: Point-based 3D single stage object detector[C], Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2020: 11040-11048.

[25] Zhou Y, Tuzel O. Voxelnet: End-to-end learning for point cloud based 3D object detection[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2018: 4490-4499.

[26] Yan Y, Mao Y, Li B. Second: Sparsely embedded convolutional detection[J]. Sensors, 2018, 18(10): 3337.

[27] Lang A H, Vora S, Caesar H, et al. Pointpillars: Fast encoders for object detection from point clouds[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 12697-12705.

[28] SShi W, Rajkumar R. Point-GNN: Graph neural network for 3D object detection in a point cloud[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 1711-1719.

[29] He C, Zeng H, Huang J, et al. Structure aware single-stage 3d object detection from point cloud[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 11873-11882.

[30] Chen Y, Liu S, Shen X, et al. Fast point R-CNN[C]. Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019: 9775-9784.

[31] Shi S, Guo C, Jiang L, et al. PV-RCNN: Point-voxel feature set abstraction for 3D object detection[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 10529-10538.

[32] Llinas J, Hall D L. An introduction to multi-sensor data fusion[C]. Proceedings of IEEE International Symposium on Circuits and Systems . IEEE, 1998, 6: 537-540.

[33] Coué C, Fraichard T, Bessiere P, et al. Multi-sensor data fusion using Bayesian programming: An automotive application[C]. Intelligent Vehicle Symposium, IEEE, 2002, 2: 442-447.

[34] Sun S L, Deng Z L. Multi-sensor optimal information fusion Kalman filter[J]. Automatica, 2004, 40(6): 1017-1023.

[35] Liu T, Du S, Liang C, et al. A novel multi-sensor fusion based object detection and recognition algorithm for intelligent assisted driving[J]. IEEE Access, 2021, 9: 81564-81574.

[36] Bewley A, Ge Z, Ott L, et al. Simple online and realtime tracking[C]. 2016 IEEE International Conference on Image Processing(ICIP). IEEE, 2016: 3464-3468.

[37] Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric[C]. IEEE International Conference on Image Processing (ICIP), IEEE, 2017: 3645-3649.

[38] Petrovskaya A, Thrun S. Model based vehicle tracking in urban environments[C]. IEEE International Conference on Robotics and Automation, Workshop on Safe Navigation, IEEE, 2009, 1:1-8.

[39] Petrovskaya A, Thrun S. Model based vehicle detection and tracking for autonomous urban driving[J]. Autonomous Robots, 2009, 26(2): 123-139.

[40] Tanzmeister G, Steyer S. Spatiotemporal alignment for low-level asynchronous data fusion with radar sensors in grid-based tracking and mapping[C]. 2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI). IEEE, 2016: 231-237.

[41] Steyer S, Tanzmeister G, Wollherr D. Object tracking based on evidential dynamic occupancy grids in urban environments[C], 2017 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2017: 1064-1070.

[42] Weng X, Wang J, Held D, et al. 3D multi-object tracking: A baseline and new evaluation metrics[C]. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2020: 10359-10366.

[43] Rusk N. Deep learning[J]. Nature Methods, 2016, 13(1): 35-35.

[44] Braverman V, Ostrovsky R. Effective computations on sliding windows[J]. SIAM Journal on Computing, 2010, 39(6): 2113-2131.

[45] Reddy B K, Bano S, Reddy G G, et al. Convolutional network based animal recognition using YOLO and Darknet[C]. 2021 6th International Conference on Inventive Computation Technologies (ICICT), IEEE, 2021: 1198-1203.

[46] Song A, Zhao Z, Xiong Q, et al. Lightweight the focus module in YOLOv5 by dilated convolution[C]. 2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA), IEEE, 2022: 111-114.

[47] Feng W Y, Zhu Y F, Zheng J T, et al. Embedded YOLO: A real-time object detector for small intelligent trajectory cars[J]. Mathematical Problems in Engineering, 2021, 2021: 1-11.

[48] Gou J, Yu B, Maybank S J, et al. Knowledge distillation: A survey[J]. International Journal of Computer Vision, 2021, 129(6): 1789-1819.

[49] Jingjing H E, Haode H U O, Xuefei G, et al. A Lamb wave quantification model for inclined cracks with experimental validation[J]. Chinese Journal of Aeronautics, 2021, 34(2): 601-611.

[50] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2018: 6848-6856.

[51] Sinha D, El-Sharkawy M. Thin mobilenet: An enhanced mobilenet architecture[C]. 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). IEEE, 2019: 0280-0285.

[52] Howard A, Sandler M, Chu G, et al. Searching for mobilenetv3[C]. Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 2019: 1314-1324.

[53] Han K, Wang Y, Tian Q, et al. Ghostnet: More features from cheap operations[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2020: 1580-1589.

[54] Xu S, Ji Y, Wang G, et al. GFSPP-YOLO: A light YOLO model based on group fast spatial pyramid pooling[C], IEEE 11th International Conference on Information, Communication and Networks (ICICN). IEEE, 2023: 733-738.

[55] Lin T Y, Maire M, Belongie S, et al. Microsoft COCO: Common objects in context[C]. 13th European Conference Computer Vision (ECCV), Springer, 2014, 5(13): 740-755.

[56] Singh G, Akrigg S, Di Maio M, et al. Road: The road event awareness dataset for autonomous driving[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, 45(1): 1036-1054.

[57] Geiger A, Lenz P, Stiller C, et al. Vision meets robotics: the KITTI dataset[J]. The International Journal of Robotics Research, 2013, 32(11): 1231-1237.

[58] Alaba S Y, Ball J E. A survey on deep-learning-based lidar 3D object detection for autonomous driving[J]. Sensors, 2022, 22(24): 9577.

[59] Bodla N, Singh B, Chellappa R, et al. Soft-NMS--improving object detection with one line of code[C]. Proceedings of the IEEE International Conference on Computer Vision, 2017: 5561-5569.

[60] GTang H, Niu X, Zhang T, et al. LE-VINS: A robust solid-state-LiDAR-enhanced visual-inertial navigation system for low-speed robots[J]. IEEE Transactions on Instrumentation and Measurement, 2023, 72: 1-13.

[61] Arunagiri P, Gnanavelbabu A. Identification of major lean production waste in automobile industries using weighted average method[J]. Procedia Engineering, 2014, 97: 2167-2175.

[62] Li T, Wang Y. Biological image fusion using a NSCT based variable-weight method[J]. Information Fusion, 2011, 12(2): 85-92.

[63] Abdi H, Williams L J. Principal component analysis[J]. Wiley Interdisciplinary Reviews: Computational Statistics, 2010, 2(4): 433-459.

[64] Naik G R, Kumar D K. An overview of independent component analysis and its applications[J]. Informatica, 2011, 35(1).

[65] You N, Han L, Zhu D, et al. Research on image denoising in edge detection based on wavelet transform[J]. Applied Sciences, 2023, 13(3): 1837.

[66] Liu X, Jiang S. Research on DS evidence reasoning improved algorithm based on Data Association[J]. TELKOMNIKA Indonesian Journal of Electrical Engineering, 2013, 11(9): 5037-5043.

[67] Sivrikaya F, Yener B. Time synchronization in sensor networks: a survey[J]. IEEE Network, 2004, 18(4): 45-50.

[68] Fu B, Wang Y, Ding X, et al. LiDAR-camera calibration under arbitrary configurations: Observability and methods[J]. IEEE Transactions on Instrumentation and Measurement, 2019, 69(6): 3089-3102.

[69] McLachlan G J. Mahalanobis distance[J]. Resonance, 1999, 4(6): 20-26.

[70] Senoussaoui M, Kenny P, Stafylakis T, et al. A study of the cosine distance-based mean shift for telephone speech diarization[J]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2013, 22(1): 217-227.