基于单目视觉的 3D 车辆检测跟踪与轨迹预测研究

周高立

2023 年 6 月

基于单目视觉的3D车辆检测跟踪与轨迹预测研究

周高立

北京理工大学

中图分类号：U469
UDC 分类号：629.4

基于单目视觉的 3D 车辆检测跟踪与轨迹预测研究

作 者 姓 名　周高立
学 院 名 称　机械与车辆学院
指 导 教 师　任宏斌
答辩委员会主席　魏巍教授
申 请 学 位　工学硕士
学 科/类 别　机械工程
学位授予单位　北京理工大学
论文答辩日期　2023 年 6 月

Research on Monocular 3D Vehicle Detection, Tracking and Trajectory Prediction

Candidate Name：	Gaoli Zhou
School or Department:	School of Mechanical Engineering
Faculty Mentor:	Hongbin Ren
Chair, Thesis Committee：	Prof. Wei Wei
Degree Applied:	Master of Science in Engineering
Major：	Mechanical Engineering
Degree by:	Beijing Institute of Technology
The Date of Defence：	June，2023

研究成果声明

本人郑重声明：所提交的学位论文是我本人在指导教师的指导下进行的研究工作获得的研究成果。尽我所知，文中除特别标注和致谢的地方外，学位论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京理工大学或其它教育机构的学位或证书所使用过的材料。与我一同工作的合作者对此研究工作所做的任何贡献均已在学位论文中作了明确的说明并表示了谢意。

特此申明。

签　　名：周高立　日期：2023.6.3

关于学位论文使用权的说明

本人完全了解北京理工大学有关保管、使用学位论文的规定，其中包括：①学校有权保管、并向有关部门送交学位论文的原件与复印件；②学校可以采用影印、缩印或其它复制手段复制并保存学位论文；③学校可允许学位论文被查阅或借阅；④学校可以学术交流为目的,复制赠送和交换学位论文；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）。

签　　名：周高立　日期：2023.6.3

导师签名：任宏斌　日期：2023.6.3

摘要

在自动驾驶系统中，感知的关键是在行驶过程中识别周围的车辆并持续跟踪，从而明确道路环境中动态的障碍车辆，除此之外对这些车辆未来的行驶轨迹进行准确估计，也能更好的帮助自车进行后续的决策规划和控制。所以本文旨在提升在驾驶视角下基于单目视觉的 3D 车辆检测跟踪与轨迹预测精度。

在单目 3D 车辆检测模型中，从主干网络提取多尺度基本特征后分别通过深度和图像编码器将其进行对应转换。其中为解决单目深度不确定性问题，设计了由高斯分布柔和化的分类问题来估计深度值。同时从车车间的关联角度出发，使用 Transformer 机制得到最终的深度特征。在图像编码器中采用多尺度可变形注意力机制提取图像特征。在解码阶段，以深度特征作为引导，与图像特征使用交叉注意力、自注意力和前馈神经网络来进行特征融合。最后通过各检测头输出目标 2D 和 3D 相关属性。所提出的模型在 KITTI 数据集中经验证对比，表明有更好的 3D 检测精度。

为了获取目标相关时序信息需要对其持续跟踪，所以进而提出了在 DeepSORT 框架下改进的多目标跟踪算法。在前述检测模型基础上，通过结合卡尔曼滤波估计它车运动和采用光流法来估计自车视角的变换的方法，预测上一帧的跟踪目标在当前帧的位置。在匹配阶段，设计基于 PAN 结构的残差卷积网络来提取外观特征，并结合由马氏距离度量的位置关系进行级联匹配，并用 GIoU 匹配进行补充。在更新阶段考虑到外观特征的时序关联性，将新特征与旧特征进行线性组合。实验证明改进的跟踪算法在跟踪精度和 ID 切换等评价指标下都有了更好的表现。

针对车辆轨迹预测问题，综合考虑了车辆特性、环境约束与车车交互行为。车辆特性是对历史轨迹采用 FPN 结构下的因果卷积来提取特征；环境约束则是采用空洞车道卷积来聚合自身、左右和前后的车道线节点特征，再通过空间交叉注意力机制赋予车辆；车车交互行为使用空间自注意力机制完成。然后使用残差全连接层分别对中点和多终点进行预测，最后将所有融合特征进行解码输出预测轨迹。在 Argoverse 轨迹预测数据集中进行测试对比，表明所提出的算法具有更好的预测效果。

本文最后将搭建的检测跟踪模型应用于实车采集的数据，表明能够实现较稳定的检测和跟踪多车辆目标。此外对多车轨迹预测进行可视化展示，也有较好的效果。

关键词：单目 3D 车辆检测；深度估计；多目标跟踪；数据关联；轨迹预测

Abstract

In the autonomous driving system, the key to perception is to detect and track the surrounding vehicles during driving, so as to identify the dynamically obstacle vehicles in the road environment. In addition, accurate estimation of the future driving trajectory of these vehicles can also be better help the self-vehicle to carry out subsequent decision-making, planning and control. Therefore, this paper aims to improve the accuracy of monocular 3D vehicle detection, tracking and trajectory prediction.

In the monocular 3D vehicle detection model, the multi-scale basic features are extracted from the backbone network and converted by depth and image encoder respectively. In order to solve the problem of monocular depth uncertainty, a classification problem softened by Gaussian distribution is designed to estimate the depth value. At the same time, from the perspective of the relationship between vehicles, the Transformer is used to obtain the final deep features. Image features are extracted by using multi-scale deformable attention in the image encoder. In the decoding stage, the depth feature is taken as the guide, and the final feature is fused with the image feature by using cross attention, self-attention and feedforward neural network. Finally, the 2D and 3D related attributes of the target are output through each detection head. After validation and comparison in KITTI data set, showing the proposed model has better 3D detection accuracy.

In order to obtain the timing information of the target, it needs to be continuously tracked, so an improved multi-target tracking algorithm based on the DeepSORT framework is proposed. On the basis of the aforementioned detection model, the position of the tracking target in the previous frame is predicted in the current frame by combining the Kalman filter to estimate the motion of other vehicles and the optical flow method to estimate the transformation of the vehicle's perspective. In the matching stage, a residual convolutional network based on the PAN structure is designed to extract appearance features, combined with the positional relationship measured by the Mahalanobis distance for cascade matching, and GIoU matching is supplemented. Considering the temporal correlation of appearance features in the update stage, the new features are linearly combined with the old ones. Experiments show that the improved tracking algorithm has better performance under the evaluation indicators such as tracking accuracy and ID switch.

For vehicle trajectory prediction, vehicle characteristics, environmental constraints and vehicle-vehicle interaction are considered comprehensively. The characteristics of vehicle

are extracted from the historical track by using the causal convolution under FPN structure. The environmental constraint is to use the dilated lane convolution to aggregate the characteristics of itself and the surrounding nodes, then integrate the vehicle features through the spatial cross-attention mechanism. The vehicle-vehicle interaction is achieved using the spatial self-attention mechanism. Then, the residual full connection layer is used to predict the midpoint and the multi-end point respectively. Finally, all the fused features are decoded and output to predict the trajectories. The results of test and comparison in the Argoverse trajectory prediction data set shows that the proposed algorithm has a better prediction effect.

At the end of this paper, the detection and tracking model is applied to the data collected by real vehicle, which shows that it can realize stable detection and tracking of multi-vehicle targets. At the same time, the visual display of multi-vehicle trajectory prediction also has a good effect.

Key Words: monocular 3D vehicle detection; depth estimation; multiple object tracking; data association; trajectory prediction

目录

第 1 章 绪论

1.1 论文研究背景和意义

近年来，随着自动驾驶技术的发展，市场对辅助驾驶系统和自动驾驶系统的需求量也越来越大，这是因为不同于人类在驾驶过程中会发生精力不集中，驾驶疲劳等错误行为[1]，计算机可以不间断地完成算法指定任务，从而也能更好的保护自身与其它参与者的安全，避免交通事故的发生。而一个完整的自动驾驶系统如图 1.1 所示，通常都离不开感知，决策规划和控制，其中感知层是对车辆所处环境进行识别和自身状态估计的环节，通过各类传感器，包括相机、激光、超声波雷达，以及自车状态相关的全球卫星导航 GNSS、惯导 IMU、轮速计等诸多设备实现环境感知和自身定位等功能。环境感知作为主要的信息输入，是进行一切后续处理的前提，只有自动驾驶系统对周边环境的理解接近甚至超过人的水平时，才能够更好地帮助无人车进行决策规划和控制，从而完成在不同交通环境下针对各种复杂工况的自动驾驶任务。所以对身处同一环境下的交通参与者进行及时有效地识别和分析是具有重要意义的研究方向。

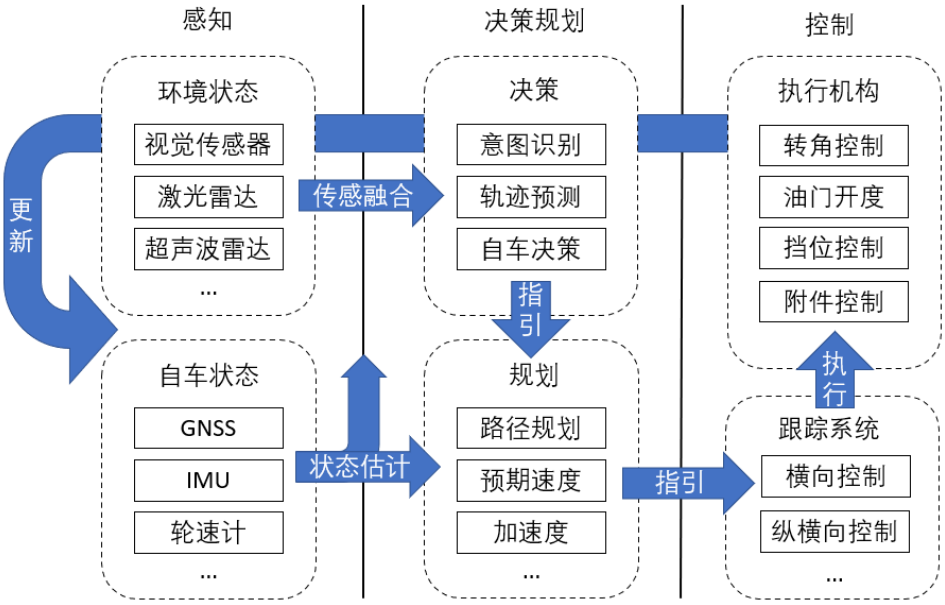


图 1.1 自动驾驶系统架构

如今感知系统的发展主要分成了两个方向，一种是冗余传感器的方向，例如 AutoX Gen5 配备了更多的雷达与摄像头，力求对自车周边环境的完全感知，提高感

知间的冗余互补；另一种则是精简传感器，压缩成本的方向，例如纯视觉方案的特斯拉的 FSD（Full Self-Driving），百度的 Apollo Lite，认为简单的视觉传感器已经可以满足感知需求，应该从算法层面提高自动驾驶能力。实际上，也由于前者所需昂贵的传感器，比如多线激光雷达等，极大的增加了硬件成本，使得自动驾驶成为少数人才能购买和享有的权利，考虑到大多数驾驶者主要依靠视听触觉，尤其是视觉，就能够完成安全驾驶行为，所以侧重提高对“有限”信息处理的算法更符合人类驾驶方式，且相对更低廉的价格也更符合市场需求，因此使用精简低廉的视觉传感设备完成自动驾驶系统的感知输入依然是主流发展方向。

视觉感知任务中最重要的部分是目标检测，即在不同的交通场景下对实时输入的视频流或每一帧图片，识别到有哪些交通参与者存在，且分别处于什么位置。不同于 2D 目标检测，识别的是目标在相机像素中的位置及大小，基于视觉的 3D 目标检测，则输出包括位置坐标，三维尺寸，方向角信息等，从而定位出目标在相机坐标系下占据的三维空间位置。基于视觉的 3D 检测方法依据设备不同也分为双目和单目，虽然双目立体视觉可以根据视差得到更好的深度估计来完成 3D 检测任务，但是考虑到在极端情况下可能发生设备同步失效或者因为某一目相机被部分遮挡导致算法整体失效等情况，所以进一步探索如何使用最简单最低廉的传感器，即单目有限感知的前提下，也能够最低限度的保障信息输入仍然有着重要现实意义。因此基于单目视觉的三维目标检测是本文的一个重点研究内容。

对于检测到的目标另一项重要任务是目标跟踪（Object Tracking），这里的跟踪是指在图像或者视频序列中能够持续地跟踪并识别到同一目标，以车辆为例，要分析其运动行为，或者预测下时刻的意图，就需要不断获取该目标的时序信息，所以需要在每一次目标检测的过程中，都能够跟踪到同一车辆对应的数据信息。

当获得车辆的时序信息后，如果直接进行决策规划是需要极高的实时处理能力，需要以当前环境状态及时调整驾驶策略。然而在实际情况中，一方面仅以感知作为输入，再进行数据处理与控制，是存在时滞影响，而对于要避免交通事故的自动驾驶系统来说是无法容忍这种时间滞后；另一方面，不考虑它车的规划，每辆车都只注重自身当下的局部最优解，这样的非全局最优的决策规划也并不有利于改善交通现状，同时因为所有交通参与者都是动态的不定因素，所以自车也很难规划出一条安全的轨迹。而处理这种问题的有效方法便是轨迹预测，预测交通参与者是否转向，未来会以怎样的轨迹行驶，这也更符合实际驾驶员决策的思考过程，从而可以不断

优化驾驶策略，更好地规划自车轨迹，同时降低未来重规划的计算负荷。如图 1.2 所示，当添加预测模块后不再需要实时改变当前规划，提高了行车的安全性以及舒适度。

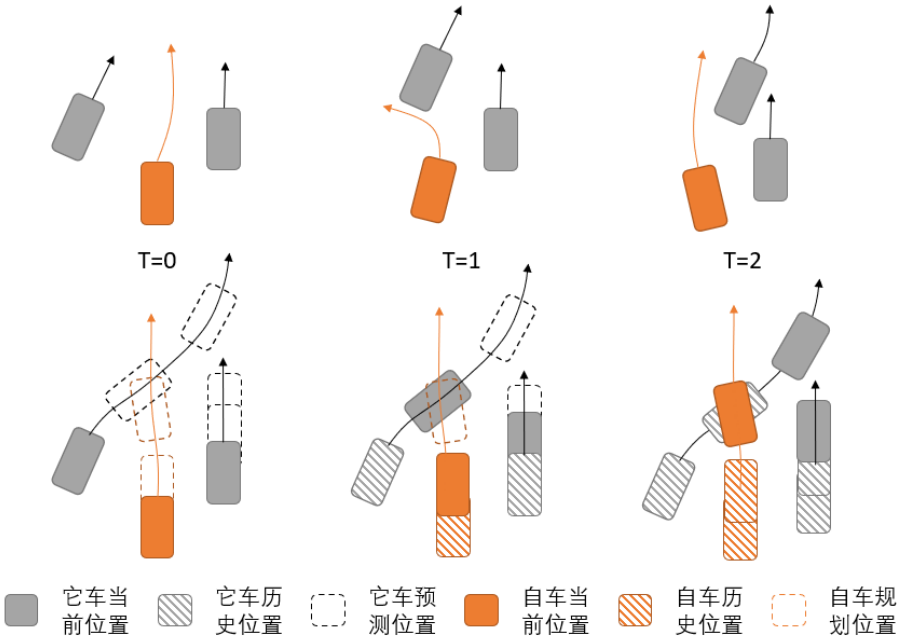


图 1.2 感知-规划示意图（上），感知-预测-规划示意图（下）

1.2 国内外研究现状及发展趋势

1.2.1 单目 3D 目标检测研究现状

3D 目标检测一直是检测领域的热点，很多成熟的检测算法多依赖激光雷达扫描的点云信息，但是所需设备昂贵，维护成本高，相比之下，单目 3D 目标检测只利用一台摄像机就可以完成目标检测任务，更加经济便捷，但是也由于单目 3D 目标检测存在深度不确定性，如图 1.3 所示，所以也极具挑战。

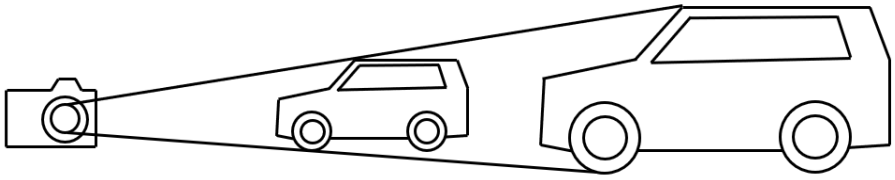


图 1.3 单目检测深度不确定性

单目 3D 目标检测算法根据主体思想的不同可以粗略的分为伪雷达法，直接回归法，深度估计三类。伪雷达方法主要由两部分组成，一是如何使用单目相机代替基

于激光雷达传感器生成的 3D 点云作为输入信息，二是使用依靠 3D 点云信息进行三维目标检测的主流雷达算法。Wang 等人[2]提出的 Pseudo-Lidar 便是采用 DORN[3]进行单目深度估计，然后将像素深度图反投影成伪 3D 点云，再利用 Frustum PointNets[4]进行三维目标检测。其中 DORN 是将深度预测作为在对数空间内的有序分类问题，而 Frustum PointNets 则是将从图像获取的 2D 检测区域与该区域的深度信息结合组成锥体候选区域（Frustum Proposal），然后进一步进行 3D 实例分割，回归三维边界框，实现三维目标检测。You 等人[5]在前者基础上提出 Pseudo-Lidar++，使用 4 线稀疏激光雷达代替 64 线进行数据修正微调检测结果。Ma 等人[6]提出 AM3D，认为前述算法是分别处理伪 3D 点云和图像数据，再简单的拼接在一起，并没有进行有效交互，所以引入注意力机制的融合方式，增强三维物体特征识别。受到伪雷达方法启发，Chen 等人[7]认为图像变点云的转化过程跨度较大，从而提出伪立体视觉法 Pseudo-Stereo 3D Detection，由左视图生成虚拟右视图，然后使用立体图像的 3D 检测算法，这也提供了另一种思路。

直接回归法是基于 2D 边界框开始，利用几何属性直接回归估计 3D 边界框。Mousavian 等人[8]提出的 Deep3DBox 网络，就是先通过二维图像目标检测确定 2D 边界框，再通过姿态回归网络求解物体的朝向及尺寸信息，而 2D 和 3D 的几何约束体现在 3D 可以被 2D 边界框紧密包围，2D 框的每条边上都能找到至少一个 3D 框的角点，最后求解目标中心到相机中心的平移关系，使 3D 边界框向底面投影中心坐标和 2D 边界框中心坐标的误差最小。Naiden 等人[9]提出 Shift R-CNN，是在前者的基础上将 2D，3D 边界框和相机参数合并起来作为输入，采用全连接网络预测 3D 位置。Brazil 等人[10]提出 M3D-RPN 网络，使用独立的三维区域生成网络，利用 2D 和 3D 透视的几何关系，经过卷积预测 3D 边界框，并使用深度感知卷积层来加深 3D 场景构建，该算法也证明了单阶段网络的有效性。Simonelli 等人[11]提出 MonoDIS 网络，针对训练过程中同时回归朝向，三维尺寸和目标中心导致损失大小不一致的问题，设计了解耦的回归损示，将回归部分分成多组来学习自身参数，从而避免不同参数之间相互依赖和误差传递干扰，从而使训练结果更稳定。Qin 等人[12]提出的 MonoGRNet 则是将 3D 目标检测任务分为四个子任务：2D 目标检测，实例级深度估计，投影 3D 中心估计和局部角点回归，将估计的中心深度结合估计水平和垂直方向的位置，实现较为准确的预测三维中心位置。受到 2D 目标检测无锚框方法的启发，Li 等人[13]提出 RTM3D，直接预测 3D 边界框的中心点和 8 个顶点，然后最小化物体

的位置、尺寸和方向在 3D 透视图中的重投影误差。Liu 等人[14]提出的 SMOKE 借鉴 CenterNet[15]的方法，直接预测 3D 边界框中心点的投影位置，再进一步回归尺寸和方向信息。Chen 等人[16]提出 MonoPair 尝试根据车辆之间的空间关系完善检测结果，通过 3D 全局优化来解决遮挡可能产生的问题。

基于深度估计方法，主要是为深度估计设计子网络来求解单目本身的深度不确定性问题。Xu 等人[17]提出的 MF3D 就是通过全卷积网络预测整张图的深度图。Zhang 等人[18]提出 MonoFlex 是将利用物体的高度比来预测其深度的方法与不确定理论相结合，完成深度估计。Ding 等人[19]提出 D4LCN，认为传统 2D 卷积不适合 3D 检测，所以提出一种新的局部卷积网络（LCN，Local Convolutional Network），命名为深度引导的动态逐深度分离局部卷积网络，其卷积核是从深度图中自动学习得到，并且单独作用在对应图片对应通道的对应像素上，而不是全局卷积核，从而处理不同大小的目标。Reading 等人[20]提出 CaDDN 将深度离散化为 LID（Linear Increasing Discretization，线性增长离散化）分类问题，以避免后续网络过度依赖深度预测准确性。

从上可以看出，单目 3D 目标检测包含了多项任务，但重点仍然是如何估计深度，并处理深度特征，对比将问题转化为伪点云和伪立体视觉的方式，依靠针对图像处理的 2D 目标检测的框架更为直接，所以构建深度估计子网络，完善特征融合是本文的研究方向之一。

1.2.2 基于视觉的目标跟踪研究现状

目标跟踪问题的基本任务是在一段视频序列里，给定一个目标，在后续每一帧都能对该目标进行持续跟踪和定位。根据目标的数量可以划分为单目标跟踪（Visual/Single Object Tracking）和多目标跟踪（Multiple Object Tracking）。顾名思义，前者因为应用场景的需求只跟踪一个目标，而后者旨在对视频中多个目标进行有效准确的跟踪。

多目标跟踪需要同时定位多个目标，根据目标初始化的方式，可分为基于检测的跟踪（DBT，Detection Before Track）和无需检测的跟踪（DFT，Detection Free Tracking），因为后者需要在第一帧时人工初始化跟踪的目标，所以大多数跟踪算法还是基于检测。此外，根据是否利用未来信息还可以分为在线跟踪（只利用历史帧）和离线跟踪（利用历史和未来帧）两类，后者可以充分利用整个时序信息，多用来

进行完整的视频处理，但是在实际应用中还是以前者的在线跟踪为主流，所以更多的研究也侧重基于检测的在线的跟踪算法。

传统的目标跟踪算法有基于卡尔曼滤波的算法，其包含预测更新两个阶段，根据运动模型从前一帧的位置预测当前帧的位置，以及对误差协方差进行估计，然后根据当前帧的实际观测值进行更新校正；基于粒子滤波算法，是对卡尔曼滤波算法中的变量必须满足高斯分布而提出，Breitenstein 等人[21]利用粒子滤波算法针对不可靠信息（未校准相机）来实现稳健跟踪；基于马尔可夫决策（MDP，Markov Decision Process）的算法，是将目标跟踪看作状态转移过程，Xiang 等人[22]具体将转移决策设计为判断新出现的对象是否为真，对象是否持续跟踪或暂时处于丢失状态，丢失状态是否被重新跟踪，这三种决策分别对应三个奖励函数，其参数通过强化学习得到。

目前更多是的基于深度学习的算法。Wang 等人[23]提出自动编码网络来提取视觉特征，然后用 SVM 执行相似度计算，用最小生成树解决关联任务的问题。Bewley 等人[24]提出的 SORT 实现了简单在线的实时跟踪，作者使用 Faster R-CNN 对行人进行检测，然后与用卡尔曼滤波预测的边界框比较 IoU 交并比来作为关系度量，最后使用匈牙利算法进行跟踪检测匹配。Bewley 等人[25]进一步使用残差卷积神经网络提取目标外观特征，将检测与跟踪间的相似度设计为边界框 IoU 和外观特征的余弦距离两者的综合度量，大大降低了 ID 切换数量。Ullah 等人[26]使用 GoogLeNet 进行特征提取。Fang 等人[27]则用 Inception 神经网络的提取视觉特征。Fu 等人[28]将匹配过程与时空关系进行融合，其最终成本矩阵作为高斯混合概率假设密度滤波器的似然值。

但是上述算法多在固定视角，比如俯视视角下进行多目标跟踪，而对于车载的感知设备，其视角会随着自车的运动而不断改变，所以完成运动补偿，并进一步提高跟踪算法的精度，是本文的研究方向之一。

1.2.3 车辆轨迹预测研究现状

通过不断的检测与跟踪可以得到不同目标的历史数据，而通过分析这些数据信息就可以去辨识其运动模型和交互行为，车辆的轨迹预测模型多是在此基础上进行实现。轨迹预测模型是无人驾驶车辆决策规划的重要前提，因为对于实际静态和动态的物体，尤其是后者，预测其在未来时间跨度中的空间位置可以使当前的路径规划更安全更平滑，从而提高后续控制的稳定性和舒适性，尤其对于需要紧急停车或

是紧急避让的情况下也能预留极为珍贵处理时间。参考 Lefevre 等人[29]的分类方法，可以将车辆的轨迹预测分为三类：基于物理学模型，基于意图模型，基于交互行为模型。

（1）基于物理学模型

物理学模型是通过计算当前时刻车辆所受到的物理学约束然后进行合理的时移推理从而预测下一时刻的车辆状态，主要是车辆运动学模型和动力学模型。常用的有恒定速度、恒定加速度、恒定转弯速率和速度、恒定转弯速率和加速度、恒定转向角和速度、恒转向角和加速度模型等。但是基于物理学的模型只考虑了当前的车辆状态的观测估计或修正，通过所建立的短时运动模型进行推导，所以在不能及时反馈修正的长时问题中，其预测结果并不可靠[30]。

（2）基于意图模型

基于意图的模型首先要解决的是从历史轨迹中推算驾驶员的行为意图或策略问题，进而在其引导下进行轨迹预测。有了意图先验，长期预测比单纯或者回归不确定性的物理学模型更准确。Yi 等人[31]使用聚类的方法，将轨迹划分为一系列原型轨迹，通过目标车辆的历史轨迹与原型轨迹的集合进行度量对比，完成潜在意图分类任务来实现轨迹预测。W. Ding 等人[32]则是将驾驶意图划分为左转、右转、停车等决策分类，进一步的作者[33]还通过设计行为意图网络得到更为的准确行为类别。Hu Y 等人[34]则考虑不同场景意图不同，设计基于语义意图的预测方法。Kumar 等人[35]提出了一种结合 SVM 和贝叶斯滤波的分层结构方法，以识别变道策略，从而获得更准确的识别结果。Schreier 等人[36]应用动态贝叶斯网络来判断驾驶策略，并利用与每个驾驶策略相对应的运动学模型来预测轨迹。Berndt 等人[37]则用隐马尔可夫结合模糊逻辑来预测驾驶员策略。

（3）基于交互行为模型

实际过程中，意图的产生和改变除了场景因素，更多的是来自于车辆之间的交互，所以基于交互行为的模型是更为主流的方向，相比前两种模型也能得到更准确的预测结果，也更具有解释性。Lefèvre 等人[38]提出基于两层隐马尔可夫模型的概率交互预测方法，通过计算多交互主体间的联合分布进行建模，然而不能精确识别它们长期依赖关系。Tablepour 等[39]基于博弈论的方法来解释深度互联交通环境的信息流动，对换道行为进行预测。Liu 等人[40]基于博弈的框架对合流行为预测，得到了较好的结果。

随着深度学习的发展，轨迹预测模型也有了新的发展。简单的意图和预测判断可以使用循环神经网络（RNN），比如 LSTM[41]和 GRU[42]实现，而叠加多层 RNN 就可以用来解决交互问题。Ding 等人[43]先使用 LSTM 编码器预测目标车辆意图，然后再将预测的意图和环境信息来生成未来轨迹，最后基于车辆交互经过非线性优化来确定初始预测轨迹。Dai 等人[44]使用两组 LSTM，一组是分别考虑目标车辆和每个相邻的车辆，另一组来对交互行为进行建模。Ding 等人[45]使用一组 GRU 编码器对目标车辆和周围车辆进行建模来实现更长时间预测。为预测多条轨迹解决多模态问题，Xin 等人[46]使用六个解码 LSTM 对应六种意图。尽管 RNN 系列网络是针对序列数据分析的主要神经网络，但在空间联系却不那么高效，所以也有将数据转换成图像进行卷积处理[47]。Lee 等人[48]使用六层卷积神经网络（CNN）从 BEV 鸟瞰视角中预测周围车辆的驾驶意图。Hoermann 等人[49]使用卷积-反卷积结构来预测 BEV 未来占用概率。Nachiket 等人[50]用一组编码 LSTM 提取每辆车的时间动态信息，而这些 LSTM 内部参数被处理成社交张量输入到卷积神经网络用来学习空间交互，最后使用六个解码器输出目标车辆六个意图下的轨迹。M. Schreiber 等人[51]用 CNN 处理不同时间帧里的 BEV 图像，然后将提取的特征序列输入到编码-解码 LSTM（Encoder-Decoder LSTM）学习时间相关性，解码后通过逆卷积过程输出成图像。

直接对图像进行卷积会丢失部分信息，所以可以直接将行驶的车辆和交互行为可以看作拓扑图中的节点与边的关系，从非欧式空间中提取数据信息，可以更专注周围车辆对目标车辆的影响，因此图卷积网络（GCN）也是轨迹预测中常用的网络。Li 等人[52]提出了基于 GCN 的 GRIP 的轨迹预测模型，该模型将每个车辆视为每个采样时间的节点，并考虑了相互作用相关因素。如果两个节点代表同一辆车，且采样时间相邻，则两个节点之间存在一条边，表示时间关系。如果两个节点同时代表两辆车，并且两辆车之间的距离小于固定阈值，则两个节点之间存在一条边，表示这些目标的空间关系和交互状态。图卷积模型的输出传递到 LSTM 编码器-解码器，以预测多个目标车辆的轨迹。Chandra 等人[53]使用双层 GNN-LSTM 结构来解决轨迹预测问题。第一层使用 LSTM 编码器-解码器来预测交通参与者的未来轨迹，第二层通过加权动态几何图网络（DGG）对交通参与者的交互相关因素进行建模。自从带有矢量地图的 Argoverse 数据集[54]提出以来，使用图神经网络（GNN）来获得车辆之间、车辆和地图之间的交互特征，进一步提高了轨迹预测的准确性。Gao 等人[55]提出的 VectorNet 以场景中的车辆和矢量地图为节点，使用 GNN 实现轨迹预测。Liang 等人

[56]使用 CNN 提取车辆特征，使用 GCN 从矢量地图中提取车道特征，然后将这两个特征结合起来进行轨迹预测。Zhao 等人[57]提出了 TNT 的目标驱动方法，该方法定义了稀疏目标终点锚点，并选择到这些终点的最佳轨迹。后作者又提出 DenseTNT[58]将目标终点的估计设计为密集锚点，并获得比 TNT 更好的结果。Gilles 等人[59]提出的 HOME 网络，首先是使用矢量车道图和车辆轨迹特征生成目标终点的概率热图，然后从中得到终点的采样点，最后生成预测轨迹。

综上所述，轨迹预测从简单的物理学模型逐步发展到融合车与环境交互，车与车交互的深度学习网络，对于如何处理提取的特征又大致分成了状态递推，直接生成轨迹，终点引导的轨迹预测三个求解思路。

1.2.4 小结

从自动驾驶系统感知层面来看，得益于深度学习的发展，目标检测，目标跟踪和车辆轨迹预测都已有一定成效，但仍有提升的空间：

（1）单目 3D 目标检测主要难点是解决深度估计，一方面借鉴现有的深度估计网络通常是将其作为一个子网络进行单独监督训练，并不是端到端的模型，该模块的输出也会直接影响后续的检测准确性；另一方面得到深度信息后再转换成伪点云或者伪立体视觉的方式都会使得特征信息转换跨度较大，所以借鉴有出色效果的 2D 目标检测框架，将深度估计的特征直接融合其中进行 3D 目标检测可能是更快速精确的算法。

（2）多目标跟踪的流程比较明确，识别目标，分析帧间目标相似度，更新迭代。为得到相似度的度量进一步分为，构建运动模型进行位置预测后，判断位置相似度，和通过外观模型提取的特征进行外观相似度判断。多目标跟踪的主要难点是解决遮挡和重现的问题，除此之外大多的跟踪视角都是固定视角，作为安置在移动车辆的相机来说如何更好的补偿前者相似度的计算还有提升空间。

（3）车辆轨迹预测，从基于物理学模型，到识别单车行为，再到融合多车交互的预测是符合实际的必然趋势，其中借助深度学习来构建历史特征和交互信息是预测的主要任务，此外预测也由单模态（只预测一条轨迹）向多模态（预测多条轨迹）发展，因为多模态的设计可以弥补单模态预测的不确定性，从而帮助后续更安全的规划。设计高精度高置信度的多模态轨迹预测模型仍是当前热点研究方向。

1.3 文章主要研究内容和技术路线

本论文旨在无人车驾驶视角环境和单目相机条件下完成针对车辆的 3D 目标检测与跟踪任务，并对后续承接的车辆轨迹预测模块进行研究。在通过大量文献调研基础上总结出可行的建模方向，并最终搭建有更高准确率的算法模型。本文的主要研究内容如下：

第二章单目 3D 车辆检测。为了从主干网络同时提取深度特征和图像特征设计了对应任务的编码器。对于深度编码器，首先采用从主干网络的最后三层的输出作为多尺度深度特征的输入，对于每一尺度特征再次经过多层卷积层提取，并在通道维度上进行叠加后用额外的卷积操作来融合不同尺度深度信息，同时本章将深度估计问题设计为由高斯分布柔和化的分类问题，来削弱因深度不确定性对后续模块的影响，并进行监督训练。在由近及远的过程中，车辆之间的深度关系通过 Transformer 机制进一步学习，并将前者的深度估计作为位置编码从而输出深度编码特征。对于图像编码器则借鉴 Deformable DETR 的 2D 目标检测方式，进行多尺度图像特征提取。在解码阶段，将深度特征作为引导，通过与图像特征间的交叉注意力、自注意力和前馈神经网络得到融合特征。最后通过各检测头输出得到 2D 和 3D 目标的位置，大小，朝向等相关属性。该章提出的模型在检测数据集中经过验证和对比，在不同指标下得到了提升。

第三章基于 DeepSORT 的多目标跟踪算法研究。本章是在第二章 3D 车辆检测的基础上，基于 DeepSORT 框架进行改进的多目标跟踪算法。首先每一帧的观测由 3D 目标检测完成，而当前帧的预测则是由卡尔曼滤波完成多目标的位置估计，同时考虑到无人车驾驶处于移动的视角，所以采用光流法对相机的运动进行补偿。在目标间的匹配阶段，基于 PAN 结构重新设计了外观特征提取网络用来建立目标的外观特征，第一次级联匹配为结合外观相似度和运动相似度的联合匹配，用来降低误匹配率，第二次匹配是针对第一次未匹配的结果，通过计算观测和预测的 GIoU 值进行二次匹配，用来提高匹配效率，具体采用匈牙利匹配算法进行数据关联。该章改进的多目标跟踪算法经由数据集验证更加有效。

第四章车辆轨迹预测。该章是针对感知后续的车辆轨迹预测问题进行进一步研究。在轨迹预测特征提取阶段，综合考虑车辆历史信息，车道环境信息，以及车车交互信息，其中历史信息采用因果卷积和 FPN 结构进行特征提取，车道约束则用 LaneGCN 的车道卷积进行特征提取，为了将车道约束赋予车辆特征，采用空间交叉

注意力机制，在此约束下，车车交互同样使用交叉注意力完成。在长时预测问题中，鉴于良好的中点和终点可以限制预测轨迹的整体误差，所以将预测的中点，和多终点处特征也赋予每一辆车中，最终使用残差全连接层对其解码得到最终预测轨迹。

第五章是对所提出的检测与跟踪模型进行实车定性分析，并将车辆轨迹预测拓展为多车辆轨迹预测进行结果可视化分析。

本文技术路线如图 1.4 所示：

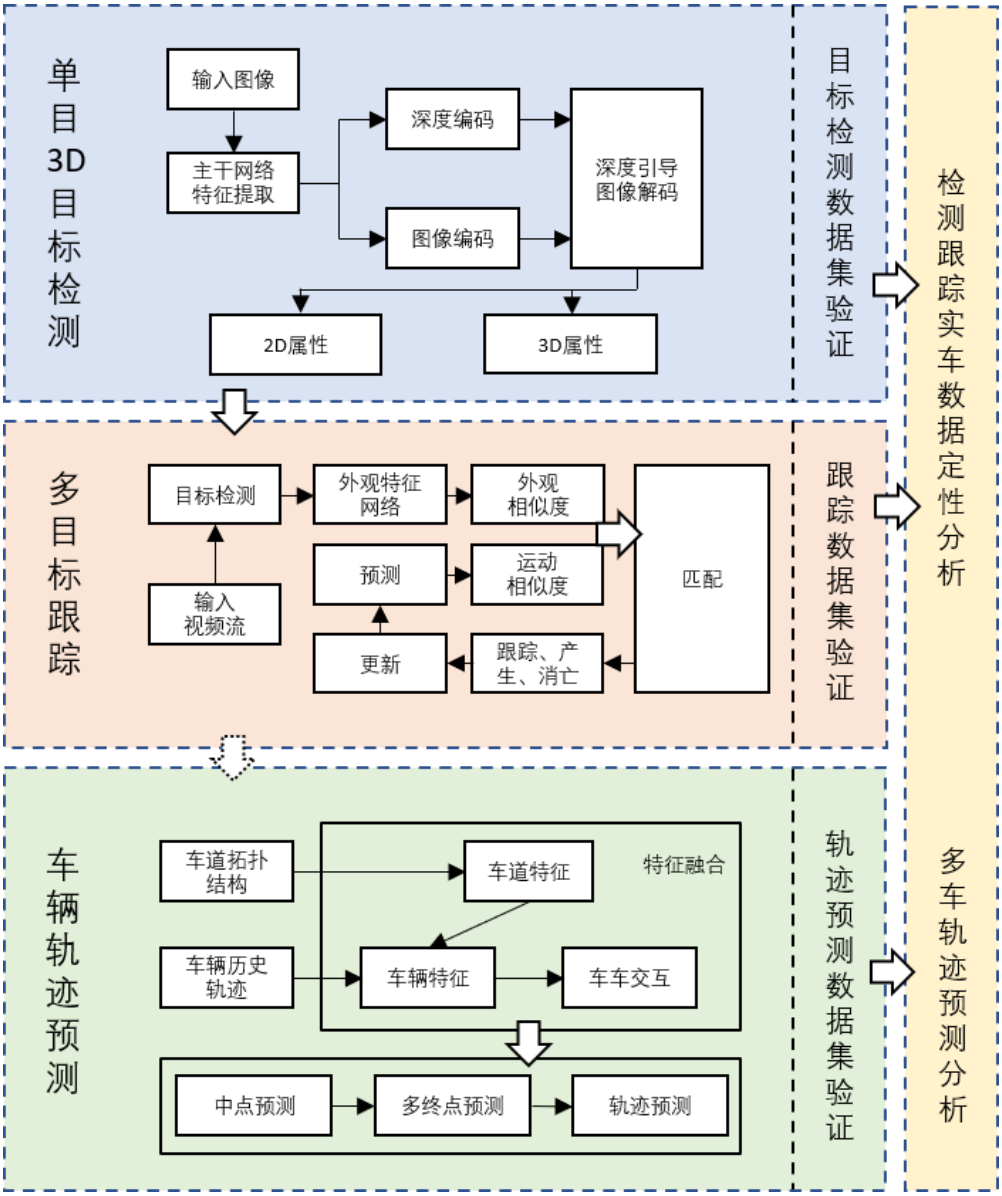


图 1.4 全文技术路线图

第 2 章 基于深度引导的单目 3D 车辆检测

2.1 引言

单目 3D 车辆检测是指通过输入的单一视角图像，除了识别出车辆及其置信度外还包括 3D 属性：相对于相机坐标系下的位置，长宽高尺寸信息，和方向，以及相对简单的 2D 属性：在像素坐标系下的位置，长宽信息，最终分别呈现为 3D 检测框和 2D 检测框。在本章中，只关注车辆一个类别，同时为了得到车辆的相对位置，设计了简单的深度估计子网络，并将深度估计转化为高斯分布柔化后的分类问题。深度编码器的设计是将深度估计值作为位置编码与深度特征通过 Transformer 机制学习深度的相对位置关系后进行合并。图像编码器则借鉴 Deformable DETR[64]进行多尺度特征提取。然后进行深度引导解码，将深度估计特征融入到图像特征中使之包含 3D 信息。最后通过各检测头输出 2D 和 3D 属性。该网络设计成端到端的神经网络，并通过验证表明得到了更好的检测效果。

2.2 基本问题描述

如图 2.1 所示，3D 属性包括（1）投影到 3D 框底面的中心点(x_{3D}, y_{3D})，如白色五角星所示；（2）为了还原其 3D 位置还需要估计该点的深度值，（3）对于 3D 物体是有方向属性的，这里假设路面较为平整，只在垂直路边方向上进行旋转，即估计车辆的方向，如绿色箭头；（4）车辆的大小尺寸，包括长宽高(L, W, H)。2D 属性则是在低面中心点的基础上估计出到橙黄色边界框的 4 个尺寸(l, r, t, d)，如红色箭头所示，从而可以确定 2D 边界框。

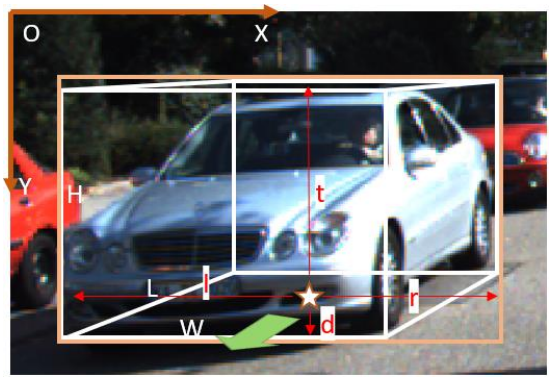


图 2.1 3D 目标检测示意图

为实现上述任务，本章设计了如下端到端的 M3DETR（Mono Depth-guided

Deformable DETR）深度估计引导的可变形 DETR 网络.

2.3 模型架构

M3DETR 的主要框架如图 2.2 所示的编码解码结构，首先由主干网络特征提取基本特征，然后对于深度估计和图像处理两个子任务设计两个编码器，包含深度估计作为前提输入的深度编码器，和图像编码器，将两部分特征进行融合与特征解码则是解码器，最后是各类检测头输出所需属性：类别，2D 属性（到边界框的长度），3D 属性（中心点位置，尺寸，朝向，深度）。

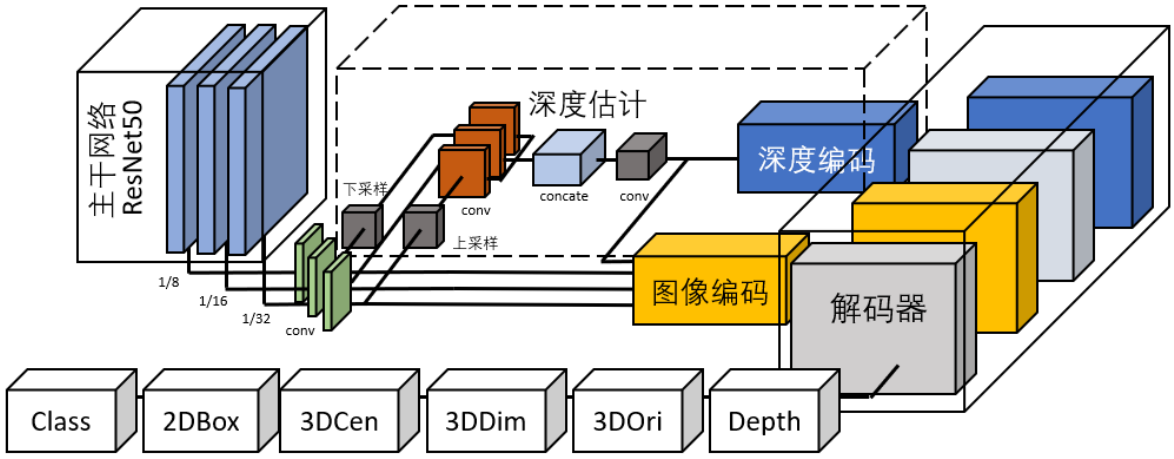


图 2.2 M3DETR 整体架构图

2.3.1 主干网络

主干网络为深度残差网络 RestNet50[60]，用来生成多尺度的深度和图像特征图。不同尺度具体反映的是神经网络在由浅及深的卷积过程中，特征图也对应从小到大的不同感受野，浅层经过的卷积较少，聚合的是局部信息，保留了更多小目标的特征，从而可以将简单目标区分开；深层经过多层卷积后，包含的则是全局信息，同时也因为分辨率的降低，小目标容易丢失，但也就更适合处理大目标，从而将复杂的目标区分开。

ResNet50 主要由卷积核为1×1和3×3的卷积层堆叠而来，依据下采样过程可以分成下表的 5 个阶段，本文采用 Layer2、3、4 的输出作为后续的多尺度特征输入。

表 2.1 ResNet50 网络结构

网络层	输出维度/(C, H, W)	卷积核/输出维度/步长
Input	(3,H,W)	-
Layer0	(64, H/2, W/2)	7×7, 64, 2
Layer1	(256, H/4, W/4)	3×3, max pool 64, 2 $\begin{bmatrix}1\times1,64\\3\times3,64\\1\times1,256\end{bmatrix}\times3$, 2
Layer2	(512, H/8, W/8)	$\begin{bmatrix}1\times1,128\\3\times3,128\\1\times1,512\end{bmatrix}\times4$, 2
Layer3	(1024, H/16, W/16)	$\begin{bmatrix}1\times1,256\\3\times3,256\\1\times1,1024\end{bmatrix}\times6$, 2
Layer4	(2048, H/32, W/32)	$\begin{bmatrix}1\times1,512\\3\times3,512\\1\times1,2048\end{bmatrix}\times3$, 2

其中$[\bullet]$表示的是 ResNet 残差单元，具体使用残差结构来解决恒等映射问题，来防止深度网络退化问题，其将输入x对应的期望输出$H(x)=x$改写为$H(x)=F(x)+x$，学习的是残差函数$F(x)$，如果该函数等于 0，就实现了恒等映射，因为拟合残差比拟合恒等映射容易，也就解决了随着网络深度加深学习效果越差的情况。

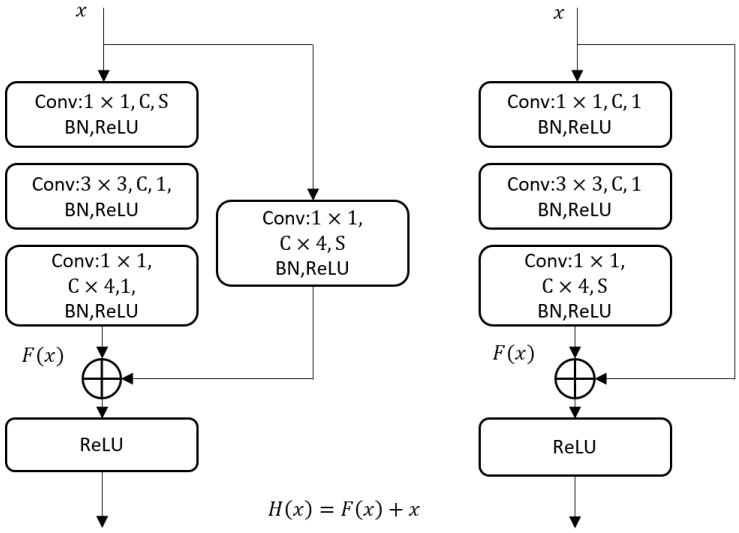


图 2.3 ResNet 残差单元

残差单元分为两种：第一种是输入与输出的通道数不相同时（如图 2.3 左），用于每一层 Layer 最开始的残差单元，在主支中，第一个卷积的步长 stride=2（对应图中 S），用以实现图像的下采样，第二个卷积采用为3×3的卷积核来聚合相邻像素信息，第三个卷积只改变了输出通道数，提升特征向量通道维度，经过图示三个卷积

后得到残差 $F(x)$，分支部分则是用来直接传递输入 x 的特征，因为通道数不同不能直接相加，所以分支部分有额外的卷积，使用同样的步长 stride=2 进行下采样使尺寸一致，同时提升通道数使与 $F(x)$ 一致；第二种是输入与输出的通道数相同时（如图右），用于每一层 Layer 的再次调用该单元时，因为第一次已经改变了通道数，所以后续的输入输出通道数相同，分支部分也不再需要额外的卷积层。

所以当输入一幅图像 $I \in \mathbb{R}^{3\times H\times W}$ 通过 ResNet50 主干网络得到了 3 层多尺度特征 $H_1 = (512, H/8, W/8)$，$H_2 = (1024, H/16, W/16)$，$H_3 = (2048, H/32, W/32)$。该三层作为提取的多尺度深度和图像特征输入到后续的深度编码器和图像编码器。

2.3.2 深度编码器

（1）深度估计网络

对于深度编码器的输入，首先要将多尺度输入特征处理为对应像素点深度的分类特征，如图 2.4 虚线框所示：

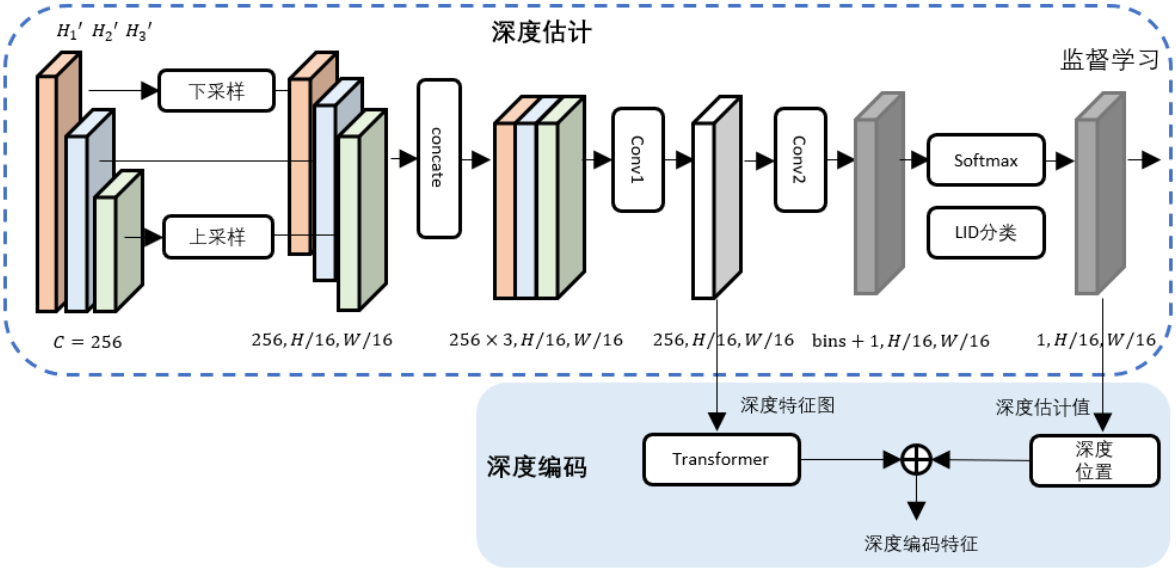


图 2.4 深度估计分类网络与深度编码

多尺度一般处理结构是 FPN[61]，由深及浅的上采样卷积，从而逐层地将深层低分辨率特征融合到浅层高分辨率特征层中，本节以图中蓝色的中间层作为基准层，来将不同 Layer 的输出特征统一到该层对应的分辨率下。具体来说首先将三层输出的特征 H_1，H_2，H_3 进行通道一致性映射，统一通道数为 256 维，得到 H_i'，然后 H_1' 的下采样由步长 stride=2 的卷积实现，卷积核设置为 3×3；H_2' 则用卷积核为 1×1，步长为 1 的卷积进行同等映射；H_3' 则需要先对特征图进行双线性插值处理得到上采样

特征图，然后再进行卷积核为1×1，步长为 1 的卷积。双线性插值原理如图 2.5，已知四个角点的位置的特征值，需要添加插入位置 p 的特征值。先在 x 方向进行线性插值得到临时特征值 $t_1(x,y_1),t_2(x,y_2)$，再在 y 轴上进行第二次线性插值得到最终 $p(x,y)$：

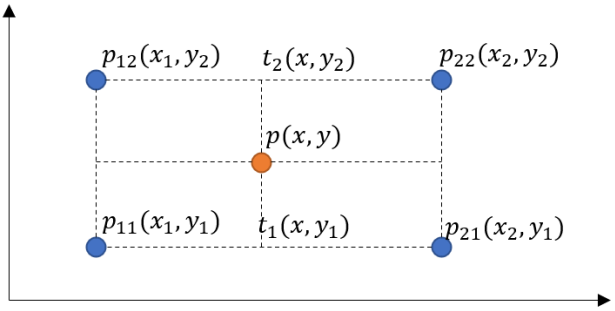


图 2.5 双线性插值原理

$$
\begin{aligned}
f(t_1) &= \frac{x-x_1}{x_2-x_1}f(p_{21})+\frac{x_2-x}{x_2-x_1}f(p_{11}) \\
f(t_2) &= \frac{x-x_1}{x_2-x_1}f(p_{22})+\frac{x_2-x}{x_2-x_1}f(p_{12}) \\
f(p) &= \frac{y-y_1}{y_2-y_1}f(p_2)+\frac{y_2-y}{y_2-y_1}f(p_1)
\end{aligned}
\tag{2.1}
$$

其中 f 表示对应位置的取值，p 对应的是所处的位置。

多尺度特征同一维度后，在通道 C 的维度上进行拼接操作，目的是将深浅特征拼接到一起，从而得到 $(256\times3, H/16, W/16)$ 的特征图，进一步的为了让网络学习到所需的合适深度特征，对通道进行压缩融合，经过 1 层卷积核为1×1，步长为 1 的卷积将 3 层信息融合得到新的特征图 $(256, H/16, W/16)$，紧接着采用两层卷积核为 3×3，步长为 1 的卷积提取深度特征。

在已有的深度特征图上去做深度估计，具体来说有两种方法，回归和分类。回归是指通过特征图直接得到预测深度，但是收敛很慢，而分类问题则是将深度区间划分为 K 个间隔（bin），预测在某个间隔中的可能性或者权重，再经过加权求和得到最终的期望深度值。

图 2.6 所示，分类方法具体主要有如下几种，UD（uniform discretization，均匀离散），SID（spacing-increasing discretization，空间增长离散），LID（线性增长离散 linear-increasing discretization），考虑到距离越远深度信息越少，预测越不准确，往往就需要区间间隔越大，提高容错率。

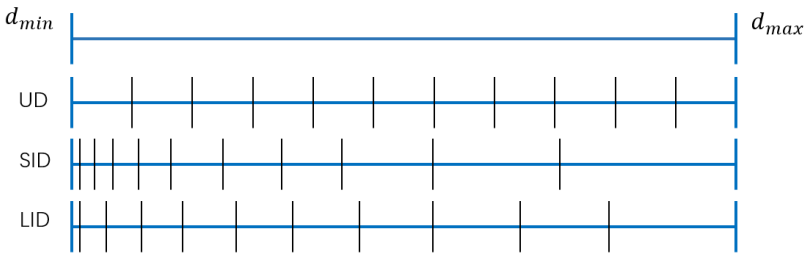


图 2.6 深度估计分类方法

参考 CaDDN[20]，本文最终采用 LID 表示的深度线性增长方法对深度间隔进行分类，公式如下：

$$\begin{cases} i_l = -0.5 + 0.5\sqrt{8(d_l - d_{\min})/\delta + 1} \\ d_l = ((2(i_l + 0.5))^2 - 1)\delta/8 + d_{\min} \end{cases} \tag{2.2}$$

其中 $d_{\max}(=60\text{m})$ 表示限定的最远深度，$d_{\min}(=0.001)$ 表示最近的深度值，$\delta = 2(d_{\max} - d_{\min})/(K(K+1))$ 表示间隔增长程度，$K(=80)$ 表示所分的间隔数。

作为监督学习的标签，深度的真值在对应的间隔（bin）的取值为 1，其余为 0，这样的处理方法过于“硬”，一旦分类不准确就会增大深度估计的误差，所以进一步的将真值标签进行高斯分布柔和，处理为如图 2.7 的软标签。这一步也主要用于后续求深度损失使用。

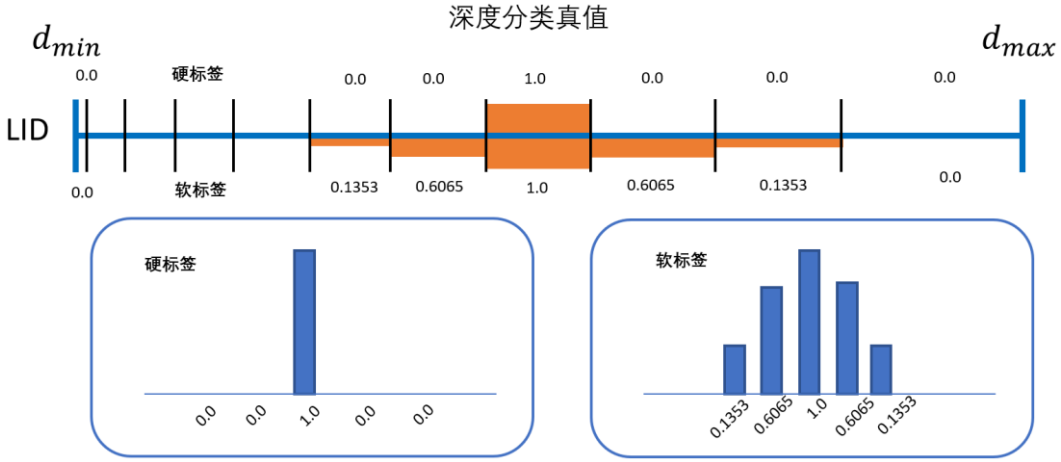


图 2.7 深度分类标签示意图

有了深度特征图和分类标准，所以网络的最后一层卷积层输出通道数为分类数的深度权重估计图（bins+1,H/16,W/16），其中“+1”是将过远的物体记为一类。最后经由 Softmax 函数统一为$[0,1]$的多分类权重。

深度分类的权重与对应分类间隔的深度值相乘求和得到最终的期望深度估计值。

$$depth = \sum depth_{weight} \odot bin_{depth} \tag{2.3}$$

式中$\odot$表示为哈达玛积(Hadamard Product)，表示的是对应元素乘积

（2）深度编码网络

在整张图片中，人类对深度远近的感受除了依靠对物体的先验尺寸与近大远小的物理规律，还有其在环境中的位置，物体与物体之间的相对位置。对于 3D 目标检测来说，前者的先验可以通过大量的数据学习得到，而相对位置的关系，或者相对远近的深度关系，则可以由 Transformer 机制来学习，其结构如图 2.8。如此更新后的特征与深度估计值结合得到最终的深度编码特征图，如图 2.4 蓝色框中所示。

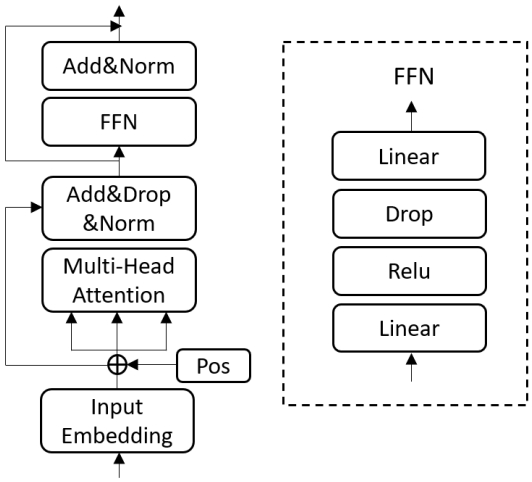


图 2.8 Transformer 结构图

Transformer 来自于谷歌提出的注意力机制[62]，Attention 注意力机制具体为如下流程，首先求解输入信息 Query（查询），与可匹配的内容 Key（索引）之间的关联程度，然后与内容本身的 Value（价值）相乘，最后得到关联的特征中最应该的关注的那部分内容：关联度越低，内容价值越低，说明 Q，K 之间越没有关系，也越无需关注，相反则说明更应关注对应的配对。

$$\begin{aligned} Q = W^Q X, K = W^K X, V = W^V X \\ \mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V \end{aligned} \tag{2.4}$$

式中 X 表示的是添加了式(2.5)位置信息的输入特征，W 为权重矩阵，d_k 为 K 的维度，Softmax 用来归一化。图 2.8 中 Multi-Head Attention 是多头注意力机制，多头体现在多个独立的 Attention 从不同任务角度并行处理，最后将输出首尾拼接，经过线性转化后得到最后特征，用以防止过拟合。

此处 Transformer 中的 Input Embedding 对应的是图 2.4 的深度特征图，Pos 位置对应的是正余弦位置编码，目的是嵌入特征图的位置信息，公式如下：

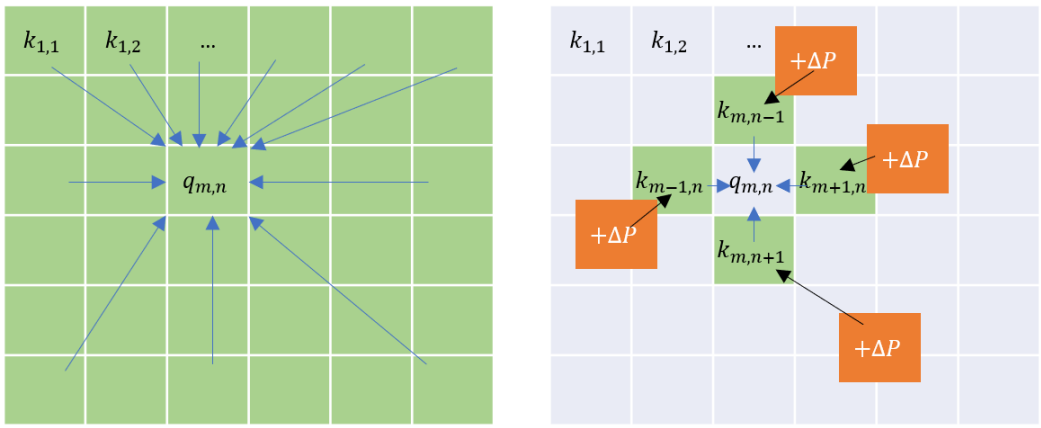
$$\begin{aligned} PE_{(k,2i)} &= \sin(k / 10000^{2i/d_{\text{model}}}) \\ PE_{(k,2i+1)} &= \cos(k / 10000^{2i/d_{\text{model}}}) \end{aligned} \tag{2.5}$$

式中 PE 表示位置编码 Position Embedding，分别指向位置 k 的编码向量的第 $2i$ 和第 $2i+1$ 个分量，d_{model} 表示的是特征维度数，10000 是自定义的常数，用来增大特征间的位置距离。因为对于图像来说有 x 和 y 两个方向，所以将上述位置编码将两个位置编码拼接在一起作为最终位置信息。

采用残差结构将多头注意力机制的输入输出相加，为提高泛化性额外增加了 Dropout 随机掩盖（mask）部分神经元进行学习。FFN 为前馈神经网络（Feed Forward Networks）其结构如图 2.8 虚线框中所示。

最后由 Transformer 更新后的深度特征图与深度估计值作为深度位置编码相加得到最终的深度编码特征。

2.3.3 图像编码器



(a)Attn 注意力　　(b)DeformAttn 可变形注意力

图 2.9 注意力(a),可变形注意力(b)示意图

图像编码器部分借鉴的是 Deformable DETR。针对 DETR[63]过长的训练时间，且在高分辨率特征图中检测小物体性能较低的两个缺点，X Zhu 等人[64]提出 Deformable DETR 进行改进，其核心思想是不再让注意力模块中的 Query 关注全局所有的 Key，如图 2.9（a）所示所有的绿色框，而是从位置上看只关注周围局部的采样 Key，如图 2.9（b）所示的绿色框作为锚框，又因为采样的 Key 并不一定正好是与 q 有关联的，

所以添加一个可学习的位置偏置 Δp，得到最终橙色框的 Key：

$$\mathrm{MultiHeadAttn}(z_q, x) = \sum_{m=1}^{M} W_m [\sum_{k \in \Omega_k} A_{mqk} \cdot W_m^{'} x_k] \tag{2.6}$$

$$\mathrm{DeformAttn}(z_q, p_q, x) = \sum_{m=1}^{M} W_m [\sum_{k=1}^{K} A_{mqk} \cdot W_m^{'} x(p_q + \Delta p_{mqk})] \tag{2.7}$$

其中(2.6)为多头注意力，m 为第 m 个 head，q 表示某一个 Query 元素，z_q 表示为其特征，同理 k 表示某一个 Key 元素，x_k 表示为对应的特征，W_m 和 $W_m^{'}$ 表示可学习的权重，A_{mqk} 表示第 m 个 head 下 q 和 k 对应的归一化注意力权重。而(2.7)可变形多头注意力的主要区别在于将 k 的范围限定到 K 个采样点，p_q 是参考点，代表 z_q 的位置，Δp_{mqk} 是相对于参考点的可学习偏移量，整个公式代表的意思是每个 q 在每个 head 中只需要和采样的 K 个位置对应的 $x(p_q + \Delta p_{mqk})$ 特征进行交互，因为位置不一定是整数，所以还需要进行双线性插值。

目标检测任务中对于大小目标的检测需要多尺度信息融合，这里与深度估计所用的多尺度相同，为同一通道数的最后三层特征输出，所以进一步的有多尺度可变形注意力：

$$MSDeformAttn(z_q, p_q, \{x^l\}_{l=1}^{L}) = \sum_{m=1}^{M} W_m [\sum_{l=1}^{L} \sum_{k=1}^{K} A_{mlqk} \cdot W_m^{'} x^l(\phi_l(p_q) + \Delta p_{mlqk})] \tag{2.8}$$

其中 $\{x^l\}_{l=1}^{L}$ 为多尺度下特征输入，l 表示不同尺度的特征图，本文分为三层，p_q 为参考点归一化 [0,1] 的坐标，这是因为不同尺度的绝对坐标相差较大不易收敛，所以需要改成归一化坐标，ϕ_l 表示映射到对应特征层。

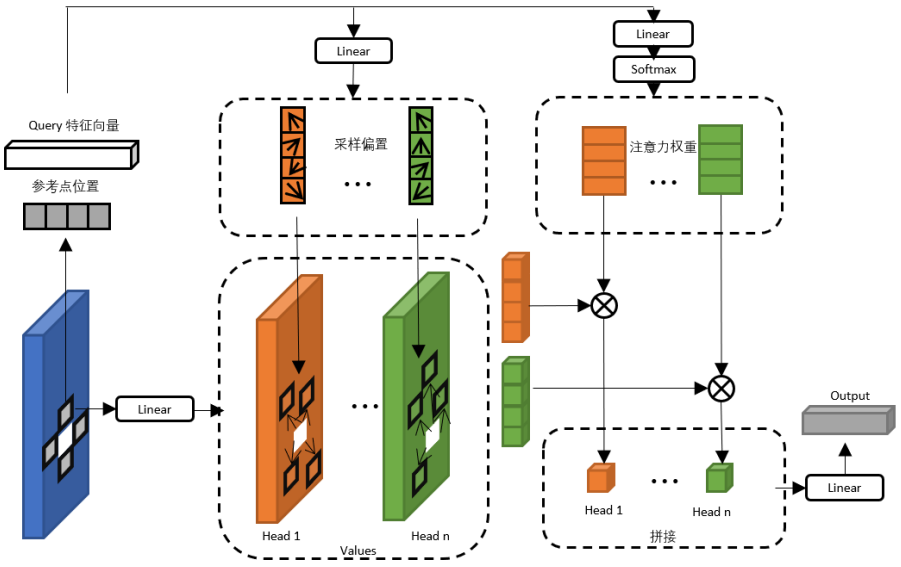


图 2.10 单尺度可变形卷积注意力

对于某一层的特征处理结构如图 2.10 所示，对于每一个 Query 选定后需找要关联的采样 Key 的参考点位置，通过对应的特征向量，一方面在采样偏置模块中学习相对参考点的偏置信息，另一方面则是在注意力权重模块中估计后续权重，Value 值则是在多个 head 中通过线性层得到（本文采用 8 个 head）。Value 与对应的权重相乘得到更新后每个 head 的特征向量，按 head 拼接经过一层线性变换得到最终的输出。

对于多尺度的输入则是每一层都做如上处理，其中的 pos 模块为两部分位置之和：图像的正余弦位置编码 Image Pos 和尺度位置 Layer Pos（1，2，3），这里添加尺度位置是为了区分不同层的输入。最终结构如下图所示：

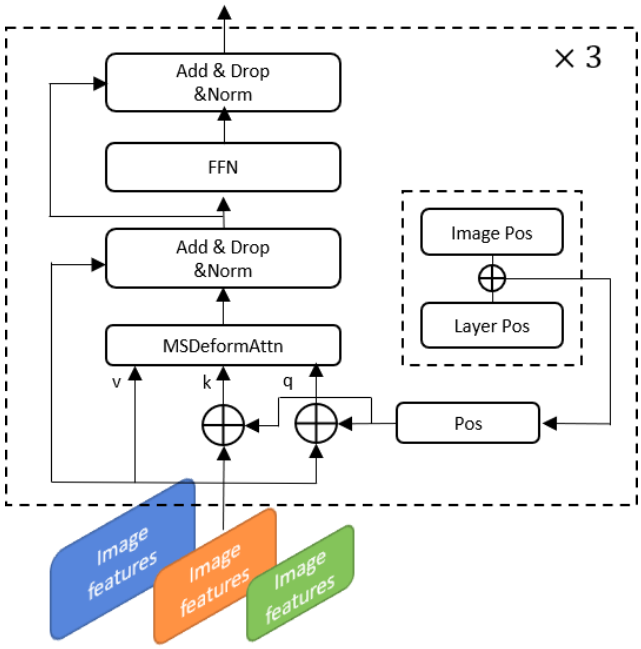


图 2.11 多尺度图像特征编码

2.3.4 深度引导解码器

如图 2.12 所示，首先构建可学习的目标序列 $q \in \mathbb{R}^{N \times C}$ 来初始化未来检测的 3D 目标，其中 N（=50）表示设定的最大检测目标数，C 对应检测的类别（本文中该值对应车辆这 1 类），因为是随机初始化，便不需要添加位置编码。而 Key，Value 则对应的是深度编码器得到的深度特征（已增加了深度估计值作为深度位置编码），如图的左侧部分，相当于是通过深度来做目标检测的预处理，预处理首先使用的 cross-attention 交叉注意力机制，交叉是因为 Query 和 Key 是来自两个不同的集合，再将融合了深度信息的目标序列输入到多头自注意力层学习内在的关联信息，而此时 Query 和 Key 是来自同一个输出的集合，此处的蓝色 Pos 位置设计为可学习的位置编码，随

机初始化后添加到 q，k 特征向量中。

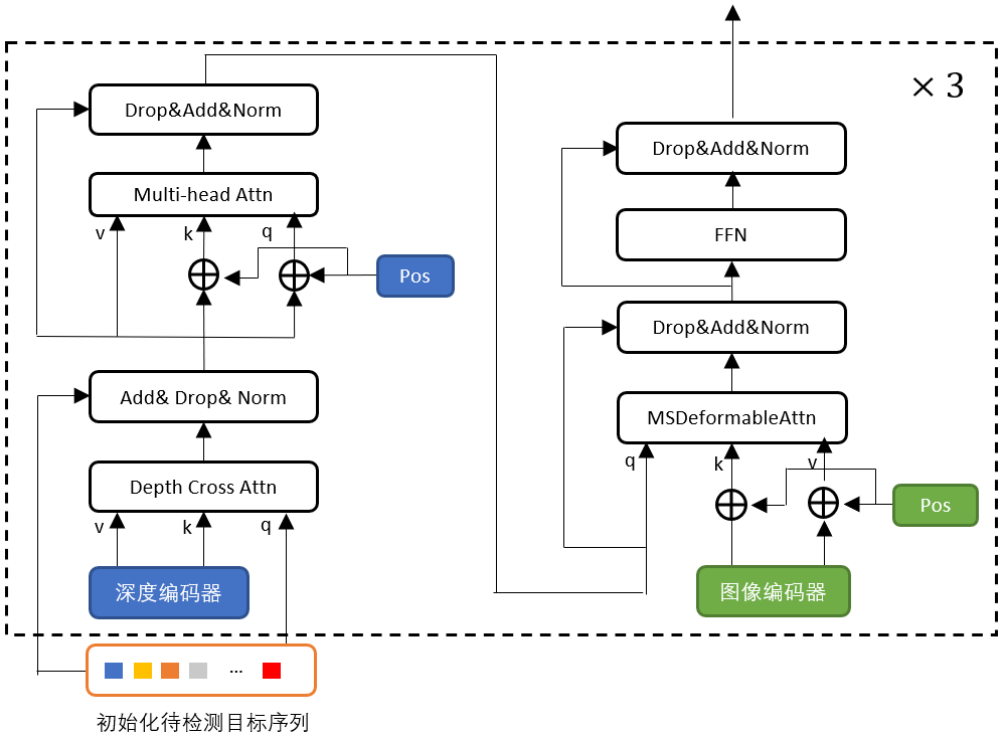


图 2.12 深度引导解码器

将深度信息与视觉信息融合则是对应上图的右侧部分，这里与图像编码器相匹配同样使用 MSDeformableAttn 模块，与图像编码器的区别在于，此时的 query 对应的是前述深度的输出，而要查询的关联对象和取值则是来自于图像编码器的输出，所以也相当于是交叉注意力机制，此处的绿色 Pos 与蓝色 Pos 为同一位置编码。

将深度与视觉特征融合后，经过前馈神经网络 FFN，再由残差结构和归一化处理得到最终的融合特征。

2.3.5 检测头及对应损失函数

经过前述的编码-解码后，输出融合后的特征，接下来分别使用不同的检测头求解物体的类别，2D 检测框的尺寸，3D 目标的投影中心，深度信息，3D 包围框的尺寸和朝向。

（1）物体的分类检测头与损失

物体分类检测头为简单的一层全连接层，输出目标序列属于类别的概率 p，因为目标序列是随机初始化，为了防止在分类过程中出现数值不稳定的情况，所以假设分类正确的先验概率为 $p_{priorty}=0.01$，从而为添加全连接层的偏置为负对数

$-\log((1-p_{priorty})/p_{priorty})=-4.595$。

常用的分类损失有交叉熵损失（CE，Cross-Entropy Loss），平衡交叉熵损示（BCE，Balanced Cross-Entropy），Focal Loss[65]等，交叉熵损失公式如下：

$$CE(p,y)=\begin{cases}-\log(p) & if\ \ y=1\\ -\log(1-p) & otherwise\end{cases} \tag{2.9}$$

其中$y=1$代表前景，其它则是背景，p代表预测的概率，为了平衡前背景的差异性，所以加上权重因子α便有了平衡交叉熵损示:

$$BCE(p,y)=\begin{cases}-\alpha\log(p) & if\ \ y=1\\ -(1-\alpha)\log(1-p) & otherwise\end{cases} \tag{2.10}$$

BCE 解决了正负样本的平衡，但没有区分难易样本（如果模型能很容易地正确分类，称为易分样本，否则为难分样本），单个易分样本的损失是低于难分样本的损失，如果易分样本的数量远多于难分样本，那么所有的样本损失也就会被大量的易份样本损失主导，所以还需要考虑难易的平衡，上式改为：

$$FL(p)=\begin{cases}-\alpha(1-p)^{\gamma}\log(p) & if\ \ y=1\\ -(1-\alpha)p^{\gamma}\log(1-p) & otherwise\end{cases} \tag{2.11}$$

其中p表示网络预测的概率，参数γ用来调节易分样本的权值比例，通常取 2。上是公式则是最终的 Focal Loss，在本节的目标分类采用同样损示:

$$L_{class}=FL(p) \tag{2.12}$$

（2）3D 投影中心检测头及其损失

参考 Monopair[67]只对 3D 投影中心点，不再对 2D 中心点进行预测，2D 边界框的位置同时由该检测头输出到边界框l,r,t,b（左，右，上，下）四个值。这里的 3D 投影中心点（Project Center）是将 3D 车辆中心点通过相机内参矩阵投影到图像平面所获得到的图像坐标。检测头采用 3 层全连接层进行预测输出，得到(x_{3D},y_{3D},l,r,t,b)

其中 3D 投影中心(x_{3D},y_{3D})与真值采用 L1 损失：

$$L_{3dproj}=\frac{1}{N}\sum_{i=1}^{N}\left|pro_{gt,i}-pro_{pre,i}\right| \tag{2.13}$$

式中$pro_{gt,i},pro_{pre,i}$分别代表真值和预测值在归一化后的图像坐标，归一化是为了解决不同图像尺寸不一致的问题，所以需要统一下$[0,1]$的尺度下。

（3）2D 边界框损失

2D 边界框损失采用 FCOS[66]所用的方法，包含两部分，一个是 3D 投影中心点到

边界框的距离，包括l,r,t,b四个值，用来确定2D包围框的位置，采用L1损失；另一个是包围框与真值的偏离程度，采用GIoU损失:

$$L_{2dbox}=\frac{1}{N}\sum_{i=1}^{N}\left|dis_{gt,i}-dis_{pre,i}\right| \tag{2.14}$$

$$L_{GIoU}=\frac{1}{N}\sum_{i=1}^{N}1-(\frac{\left|S_{pre,i}\cap S_{gt,i}\right|}{\left|S_{pre,i}\cup S_{gt,i}\right|}-\frac{\left|A_i-S_{pre,i}\cup S_{gt,i}\right|}{\left|A_i\right|}) \tag{2.15}$$

其中(2.14)表示的是真值和预测的l,r,t,b之差；(2.15)中括号内的第一项是IoU交并比损失，第二项中A_i表示的是预测框与真实框的最小闭包区，如下图所示：

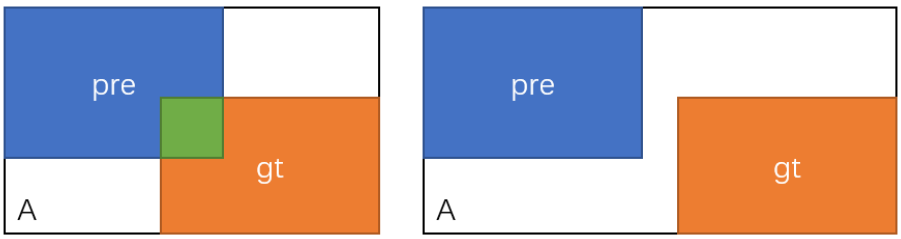


图2. 13 GIoU示意图

IoU中$\left|S_{pre,i}\cap S_{gt,i}\right|$表示的是预测框与真值框的交集，绿色部分，$\left|S_{pre,i}\cup S_{gt,i}\right|$表示的是预测框与真值框的并集，为蓝色，橙色和绿色面积的总和。IoU损失的局限性在于并不能界定如右图所示的分离情况，当没有交集时，IoU总为定值，从而不能量化预测框与真值框之间的距离，GIoU则是加入了最小闭包区，如图所示黑色框包括的总面积A，(2.15)中括号内的第二项则是判断两者的距离。

（4）辅助的深度监督损失

在深度编码器部分的输入中，设计了深度估计网络，并将其作为深度引导输入到后续模块中，辅助深度监督损失就是对这部分内容进行监督学习。因为标签只有目标车辆的位置信息，而关注的也是车辆之间的相对深度关系，所以，从真值2D边界框中对深度估计图进行mask，只对目标车辆的深度进行分类估计。其中GFL表示的是高斯分布柔和化标签的Focal Loss：

$$L_{depth'}=\frac{1}{N}\sum_{i=1}^{N}[\frac{1}{W_FH_F}\sum_{u=1}^{W_F}\sum_{v=1}^{H_F}GFL(d_{pre',i}(u,v))] \tag{2.16}$$

式中(u,v)为像素坐标，W_FH_F为像素宽度和高度，用以归一化。

（5）深度检测头及损失

最终的深度值由两部分组成，一是采用预测视差的方式，从而实现小深度值的稠密预测和大深度值的稀疏预测，具体通过两层全连接层输出得到深度的倒数和方差值；二是前述的深度估计网络预测的深度值：

$$d_{pre}=\left(\frac{1}{sigmoid(d_{reg})+\varepsilon}-1+d_{cls}\right)/2 \tag{2.17}$$

式中d_{reg}为预测的倒数值，通过 sigmoid 函数将其限定在[0,1]区间，ε防止取零值，“-1”使深度区间为大于 0 的数，d_{cls}为（2.2）通过分类问题计算得到期望深度值，两者取平均为最终的深度预测值。

深度损失采用的是 L1 对应的拉普拉斯概率不确定性损失（Laplacian Aleatoric Uncertainty Loss）：

$$L_{depth}=\frac{1}{N}\sum_{i=1}^{N}\left(\frac{\sqrt{2}}{\sigma_i}\left|d_{pre,i}-d_{gt,i}\right|+\log(\sigma_i)\right) \tag{2.18}$$

具体来说如果误差以高斯分布来界定不确定性，最大化其似然估计相当于最小化损失，从而有：

$$\begin{aligned}&\max\sum_{i=1}^{N}\log\left(\frac{1}{\sqrt{2\pi}\sigma_i}\exp\left(-\frac{\left|d_{gt,i}-d_{pre,i}\right|^2}{2\sigma_i^2}\right)\right)\\&=\max\sum_{i=1}^{N}-\frac{\left|d_{gt,i}-d_{pre,i}\right|^2}{2\sigma_i^2}-\log(\sigma_i)-\frac{\log(2\pi)}{2}\\&\Leftrightarrow\min\sum_{i=1}^{N}\frac{\left|d_{gt,i}-d_{pre,i}\right|^2}{2\sigma_i^2}+\log(\sigma_i)\end{aligned} \tag{2.19}$$

这里误差项对应的是平方项所以是 L2 损失，而 L2 范数会将误差平方化（如果误差大于 1，则误差会放大很多，对于深度估计的不确定性来说更是如此），模型的误差就会比 L1 范数大的多，因此模型会对这种类型的样本更加敏感，这就需要调整模型来最小化误差。但是很大可能这种类型的样本是一个异常值，模型就需要调整以适应这种异常值，那么就会导致训练模型的方向偏离目标。所以改为用 L1 对应的拉普拉斯分布来界定深度的不确定性：

$$\begin{aligned}
&\max\sum_{i=1}^{N}\log(\frac{1}{2b_i}\exp(-\frac{\left|d_{gt,i}-d_{pre,i}\right|}{b_i}))\\
&=\max\sum_{i=1}^{N}-\frac{\left|d_{gt,i}-d_{pre,i}\right|}{b_i}-\log(b_i)-\log(2)\\
&\Leftrightarrow\min\sum_{i=1}^{N}\frac{\left|d_{gt,i}-d_{pre,i}\right|}{b_i}+\log(b_i)
\end{aligned}\tag{2.20}$$

对比高斯分布的方差为σ^2，而拉普拉斯分布的方差则是$2b^2$，b为尺度参数，考虑到检测头输出的高斯分布方差值，所以将(2.20)对比改写为：

$$\begin{aligned}
&\min\sum_{i=1}^{N}\frac{\left|d_{gt,i}-d_{pre,i}\right|}{\frac{\sigma_i}{\sqrt{2}}}+\log(\frac{\sigma_i}{\sqrt{2}})\\
&=\min\sum_{i=1}^{N}\frac{\sqrt{2}\left|d_{gt,i}-d_{pre,i}\right|}{\sigma_i}+\log(\sigma_i)-\log(\sqrt{2})\\
&\Leftrightarrow\min\sum_{i=1}^{N}\frac{\sqrt{2}\left|d_{gt,i}-d_{pre,i}\right|}{\sigma_i}+\log(\sigma_i)
\end{aligned}\tag{2.21}$$

由此推导得到(2.18)公式，作为估计深度的 L1 概率不确定性损失

（6）3D 边界框检测头及损失

3D 边界框检测头采用 3 层 MLP 进行预测输出的得到(h,w,l)。3D 边界框损失采用 L1 损失：

$$L_{\text{dim}}=\frac{1}{N}\sum_{i=1}^{N}\left|\frac{\text{dim}_{pre,i}-\text{dim}_{gt,i}}{\text{dim}_{gt,i}}\right|\tag{2.22}$$

（7）方向损失

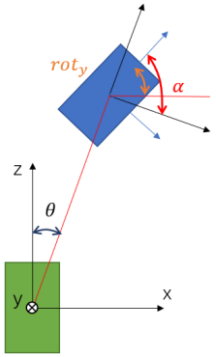


图 2.14 方向角度关系示意图

这里的角度定义如图 2.14 所示，绿色方框为自车方位，并建立对应的相机坐标

系，蓝色为被观察的目标，θ表示的是相机观察到目标中心所处的方位；α表示的是目标观测角，是目标相对于相机视角的朝向，顺时针为正；rot_y表示的目标朝向角，是目标在现实世界的朝向，顺时针为正。

从而有如下关系：

$$rot_y = \theta + \alpha \tag{2.23}$$

θ可以由相机内参和目标坐标得到：

$$\theta = \arctan(\frac{x}{z}) = \arctan(\frac{x'}{f}) \tag{2.24}$$

式中z，x分别表示在相机坐标系下目标所在的位置，f为相机的焦距，x'为在x光心处的投影。

所以只需要再预测到α目标观测角，就能计算得到方向rot_y。为了得到更为精确的α角度，检测头通过 2 层全连接层输出得到 12 个分类概率和 12 个补偿角度值，从而估计的角度是在粗略的分类角度下进行更细化的角度补偿。

α角度的损失包含两部分，一个是分类对应的交叉熵损失，将 360°分为 12 个间隔（对应的是间隔中心值），预测在某个间隔的概率$p_{pre,i}$，另一部分是残差回归，残差res对应分类后的角度补偿。

$$L_{angle} = \frac{1}{N}\sum_{i=1}^{N}(CE(p_{pre,i}) + |\, res_{gt,i} - res_{pre,i}\,|) \tag{2.25}$$

（8）总损失

最终的总损失设计如下：

$$\begin{aligned} L = {} & 2\times L_{class} + 10\times L_{3dproj} + 5\times L_{2dbox} + 2\times L_{GIoU} \\ & + L_{depth'} + L_{depth} + L_{\dim} + L_{angle} \end{aligned} \tag{2.26}$$

式中对 3D 投影中心和及其到 2D 边界框的距离权重较大，这样可以加强图像 2D 属性的学习，保证 2D 边界框预测的准确率，再不断由深度信息得到 3D 属性。

需要说明的是通过解码器调整输入的初始目标序列后，会输出后同样长度的预测目标序列，假设实际有 N 个目标真值，那么理论上应该最多只有 N 个预测目标能匹配上真值，且不会重叠，其它未匹配上的记为无目标类别，也就不需要 NMS 非极大值抑制处理。

在本文的匹配中采用和DETR[63]相同的二分匹配策略，匈牙利匹配。首先从模型输出中获取预测目标边界框的类别、位置、中心点等信息，然后与目标真值边界框

进行匹配计算损失值。具体来说，首先将预测的类别概率值进行 sigmoid 函数处理，然后使用式（2.12）Focal Loss 的方法计算分类损失。然后计算预测边界框和目标真值边界框之间的3D投影中心点的L1偏差，和该点到2D边界框左右上下的L1偏差。最后计算预测边界框和目标边界框之间的 GIoU 距离来度量它们之间的重叠情况。将上述四种损失加权求和得到总的损失值，然后使用匈牙利算法进行匹配，找到最优匹配方案。

2.4 实验结果与分析

本文采用实验室的深度学习服务器平台进行实验，该平台搭载了 Intel Xeon(R) Silver 4210 2.20-GHz 处理器，和 NVIDIA GeForce RTX 2080Ti 显卡，并以 Ubuntu 20.04 LTS 作为操作系统。

训练过程采用端到端的方式，使用 Adam[68]优化器，初始学习率设置为 1e-4，总迭代轮数为 175，并在第 145 和第 160 轮时学习率衰减为之前的 0.1，受限于显存大小，训练将 Batch size 设置为 8，

2.4.1 实验数据集

本章采用 KITTI 数据集[69]进行验证。该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办，是现阶段国际上具有代表性的在自动驾驶场景下的视觉算法评测数据集，可用于 3D 目标检测、目标跟踪等多个任务的学习。其数据采集平台配有两个灰度摄像机（FL2-14S3M-C），两个彩色摄像机（FL2-14S3C-C），一个 Velodyne 64 线 3D 激光雷达，4 个光学镜头和 1 个 GPS 导航系统。

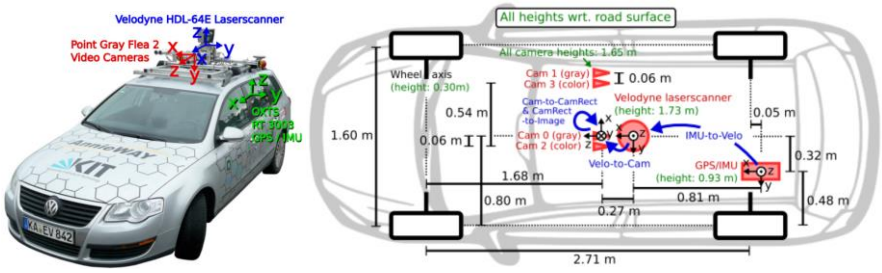


图 2.15 KITTI 数据集采集平台及传感器布置

考虑到 KITTI 目标检测集没有可供使用的测试集标签，所以参照 Chen 等人[70]的处理方式，将训练集进一步分为包含 3712 张图像和标签的子训练集和包含 3769 张图像和标签的验证集。标签释义如下表：

表 2.2 KITTI 目标检测数据释义

字段	字段长度	单位	含义
Type	1		目标类别
Truncated	1		因图像尺寸原因导致的截断程度
Occulded	1		遮挡程度：0：完全可见；1：部分遮挡；2：大部分遮挡；3：未知
Alpha	1	弧度	目标观测角[-pi,pi]
Bbox	4	像素	目标 2D 检测框位置：左上和右下两坐标
Dimension	3	米	目标 3D 尺寸高，宽，长
Location	3	米	3D 框底面中心(x,y,z)，处于相机坐标系
Rot	1	弧度	目标朝向角[-pi,pi]

2.4.2 评价指标

（1）平均准确率 AP（Average Precision）

该评价指标的前提条件是判断目标是否被正确检测出来，就需要比较检测框和真值边界框的 IoU 交并比，即两框的重叠程度，该值越大说明重合程度越高，当大于一定阈值（本文中设置为 0.7）才说明目标被检测到。对于不同检测的角度又可以细分为如下三种情况，$AP|_{3D}$ 表示的是 3D 检测框的体积交并比大于 0.7 时的 AP 值，$AP|_{BEV}$ 则是 3D 检测框投影到俯视视角下的检测框所对应的 AP 值，$AP|_{2D}$ 表示的是 2D 检测框面积交并比符合条件时的 AP 值，前两种为 3D 检测主要评价指标，最后一种为 2D 检测的评价指标。

筛选对应的检测框后，其类别问题通常使用准确率（Precision）和召回率（Recall）来衡量分类算法的性能。准确率表示的是预测正确的正样本的准确度，该值越大说明误检验越少；召回率表示的是预测正确的正样本的覆盖率，该值越大说明漏检的越少。平均准确率 AP 是在多个召回率阈值下求得准确率的平均值：

表 2.3 样本分类结果

真实标签	预测为正(Positive)	预测为负(Negative)
True	TP(True Positive)	FN(False Negative)
False	FP(False Positive)	TN(True Negative)

$$Pre = \frac{TP}{TP + FP} \tag{2.27}$$

$$Rec = \frac{TP}{TP + FN} \tag{2.28}$$

$$AP = \frac{1}{N}\sum\nolimits_{r \in R_N} \max_{r' \geq r} Pre(r') \tag{2.29}$$

其中 $R_N = \{1/40, 2/40, ..., 1\}$ 共计 $N = 40$ 召回率点，遍历每个召回率点 r，取大于等于当前召回率点的所有召回率点对应的准确率中的最大值进行求和，最后再除以召回点数量得到 AP 值。

（2）平均朝向角相似度 AOS(Average Orientation Similarity)

AOS 表征的是朝向角预测指标，其定义与 AP 相似

$$OS(r) = \frac{1}{|D(r)|}\sum\nolimits_{i \in D(r)} \frac{1 + \cos\Delta_\theta^{(i)}}{2}\delta_i \tag{2.30}$$

$$AOS = \frac{1}{N}\sum\nolimits_{r \in R_N} \max_{r' \geq r} OS(r') \tag{2.31}$$

式中 $OS(r)$ 表示的是召回率为 r 的方向相似度，$D(r)$ 表示在召回率为 r 所有预测为正样本的集合，$\Delta_\theta^{(i)}$ 为召回率为 i 时，预测朝向角与真实值朝向角的差值，δ_i 用来惩罚多个预测匹配到同一个真实值，如果已经匹配到则 $\delta_i = 1$，否则为 0

依据检测的难易程度在上述基础之上又可以细分为简单（Easy）、中等（Moderate）、困难（Hard）三个等级，分别根据标注框的像素高度、被遮挡程度、截断程度进行定义，如表 2.4 所示：

表 2.4 检测难易等级定义

等级	最小边界框高度	最大遮挡	最大截断
简单	40 像素	完全可见	15%
中等	25 像素	部分遮挡	30%
困难	25 像素	难以看到	50%

2.4.3 实验结果与对比

（1）实验结果可视化

下图所示为在 KITTI 数据集典型样例图片中，使用所搭建的神经网络模型对车辆检测的 2D、3D 和 BEV 视角的检测框可视化结果。

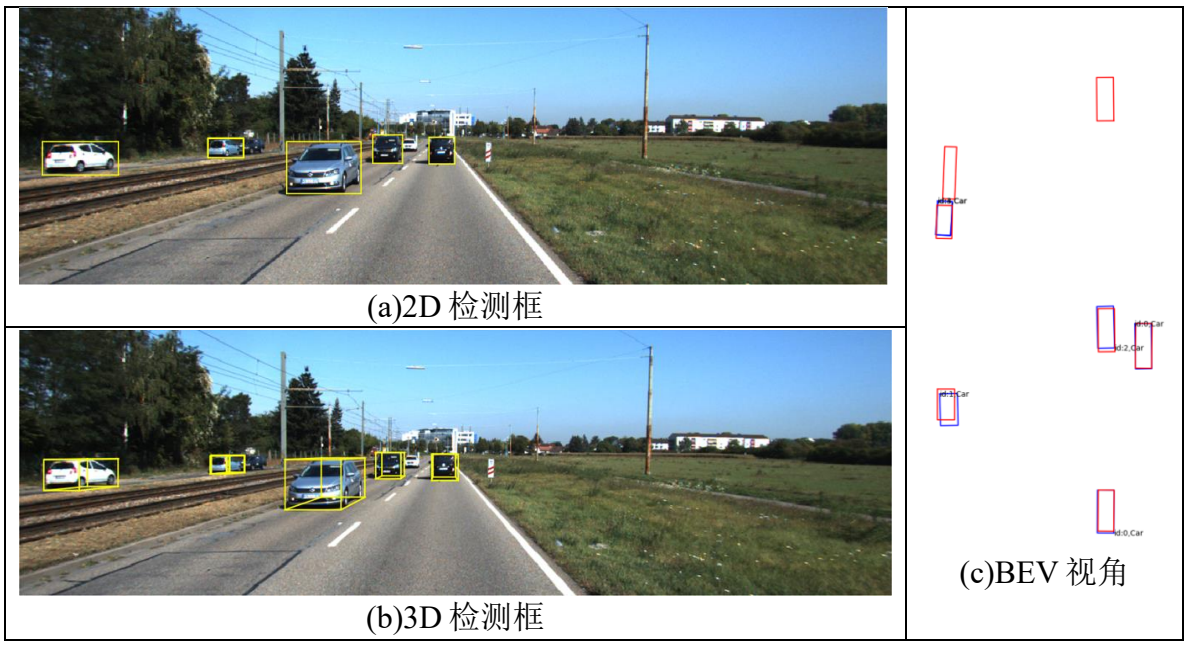


图 2.16 检测结果可视化实例一

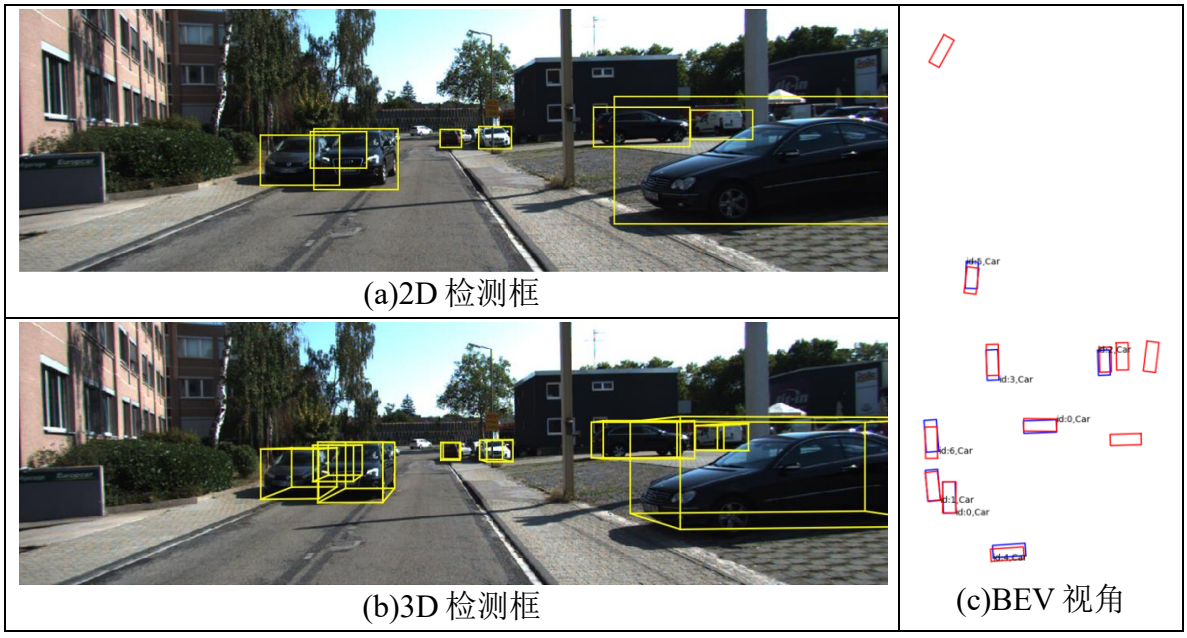


图 2.17 检测结果可视化实例二

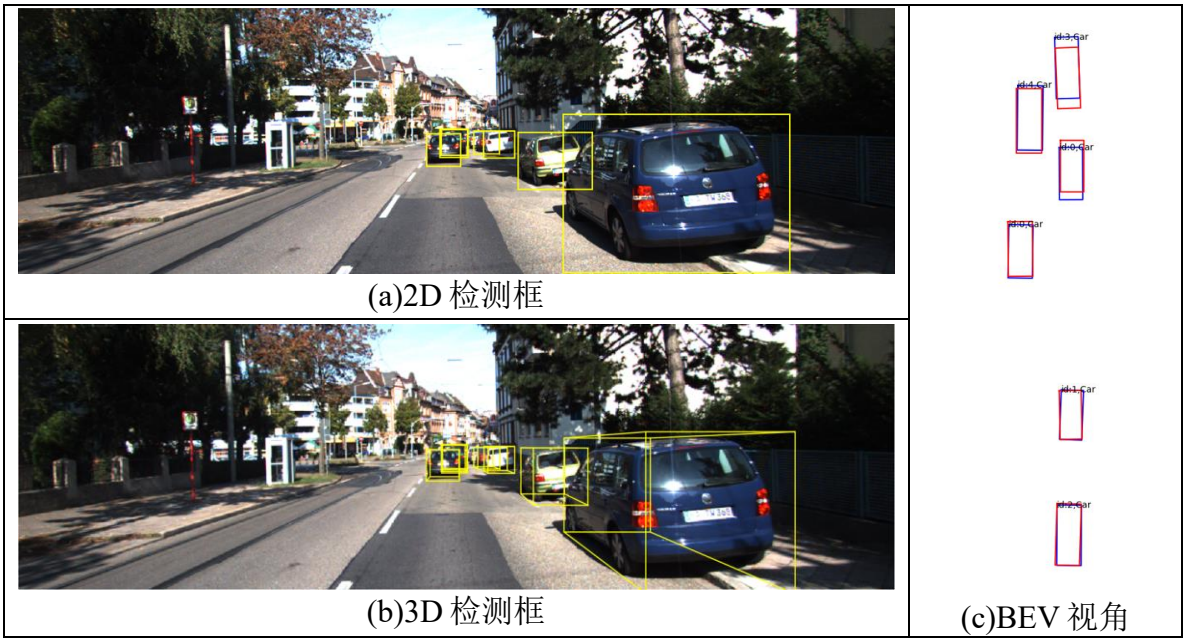


图 2.18 检测结果可视化实例三

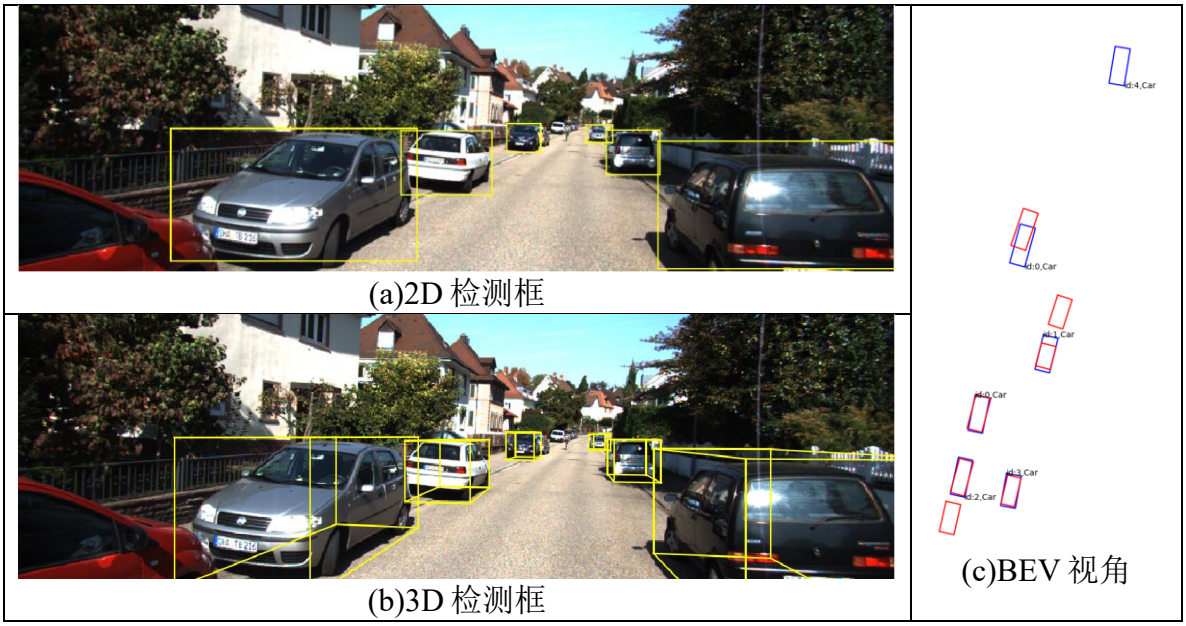


图 2.19 检测结果可视化实例四

图 2.16-19 为车辆的单目检测结果，（a）为输出的 2D 检测框；（b）为输出的 3D 检测框；（c）为转换为 BEV 视角下的检测框，其中蓝色框为预测值，红色框为真值。可以看出在不同环境下，包括公路（图 2.16），半车道与半停车场（图 2.17），城镇车道（图 2.18），和较为拥挤的车道（图 2.19）都能够较为准确的识别出车辆的位置，三维尺寸，以及方向。

从图 2.17 右方横停车辆和图 2.19 右侧车辆可以看出，对于在截断较小的情况下依然可以完成识别任务，但是当截断过大时，如图 2.19 左侧红色车辆，则识别较为困难。如图 2.18 中，在车辆遮挡不严重的情况下能保持较高识别率，但在图 2.19（c），右侧远处第三辆车因视角原因被全部遮挡，则检测较为困难。

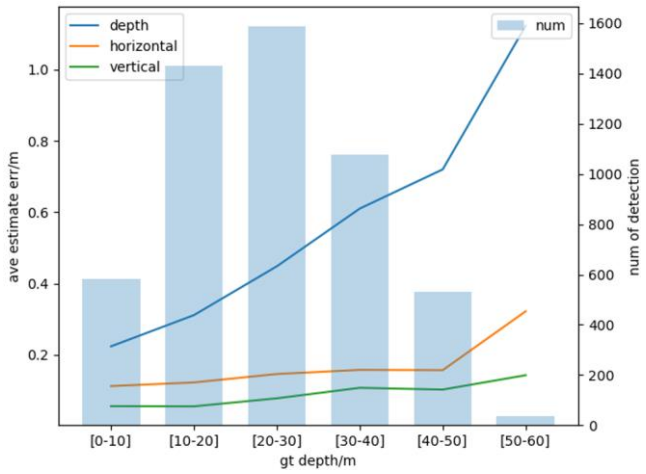


图 2.20 不同深度范围物体数量及深度估计误差

对于已明确与真值框匹配的检测框，从图 2.20 可以看出，随着车辆目标越来越远，深度估计的不准确性也越来越大，但在垂向和水平方向的误差则相对较小。同时因为远处识别目标也只占总体的极少数，所以模型拥有较好的的深度估计性能。

（2）实验结果对比

使用介绍的深度学习服务器平台，在 KITTI 验证集上依据各评价指标，对多种算法重新训练后检测结果进行量化对比，如下表所示：

表 2.5 KITTI 验证数据集检测结果

方法	AP|3D			AP|BEV		
	Easy	Mod	Hard	Easy	Mod	Hard
CenterNet[15]	0.86	1.06	0.66	3.91	4.46	3.53
Mono3D[71]	2.53	2.31	2.31	5.22	5.19	4.13
SMOKE[14]	14.76	12.85	11.50	19.99	15.61	15.28
MonoPair[16]	16.28	12.30	10.42	24.12	18.17	15.76
MonoDLE[72]	17.45	13.66	11.68	25.24	20.37	17.82
MonoDETR[73]	22.51	16.29	13.21	**33.62**	**23.58**	19.65
M3DETR(proposed)	**23.39**	**16.61**	**13.52**	32.61	23.42	**19.67**

从表 2.5 可以看出所提出的检测模型，在综合考虑了 3D 位置、尺寸、朝向的 3D 检测精度指标 $AP|_{3D}$ 中对易识别的车辆检测提升了 0.88%，中等难度的目标检测提升了 0.32%，困难程度的提升了 0.29%，说明模型对深度和图像特征融合后的综合性能表现更好；而在代表深度和水平位置、长宽和朝向的BEV检测精度 $AP|_{BEV}$ 中表现一般，是因为将较为复杂的深度估计子网络融入进端到端的模型中，受到了后续图像特征与融合的影响，在总损失反向传播中一定程度上削弱了前者的收敛效果。但结合两者可以看出所提出的深度估计分类柔和的方法确实减轻深度不准确性对后续 3D 检测精度的影响。

此外本文提出的深度估计网络中对多尺度输出特征的处理方式是在特征通道维度上拼接后进行卷积操作（记为Conv），为了验证其有效性，额外对比其他处理方式，如果采用 Transformer 机制对融合后的特征先进行深度相对位置学习再进行分类估计（记为 Trans），或是使用 CSA[74]模块进行融合学习效果都出现不同程度地降低。此外在主干网络中，如果采用 ASPP(空洞空间金字塔结构)或是 SPPF（快速空间金字塔）模块对 ResNet50 输出的最深层特征图从空间层面进行进一步特征提取，也出现收敛速度慢，检测效果也不够好，如下表所示：

表 2.6 深度估计不同操作对照检测结果（未考虑分类柔化时）

方法	AP|3D			AP|BEV		
	Easy	Mod	Hard	Easy	Mod	Hard
Conv	**22.80**	**16.10**	**13.26**	**32.25**	**22.38**	**18.64**
Trans	12.27	8.78	5.52	16.34	11.1	8.90
CSA	16.45	12.35	11.21	18.56	15.92	14.13
ASPP	14.78	12.30	10.42	19.12	17.53	15.02
SPPF	15.14	13.50	11.35	18.52	16.43	14.18

分析上述产生的原因一是深度估计中更复杂的结构增加了额外学习的参数量，使得在所提出的端到端的检测网络中对应的任务损失下，复杂的前置深度估计网络的收敛变缓，收敛不平衡；二是对于新主干网络学习用的数据量不够，且最深层的特征尺寸较小，不适合再进一步进行空间特征提取。但是本文最后提出的深度估计还是未采用 MonoDETR[73]对多尺度特征取平均的方式，因为平均并不能代表多尺度下深度特征的均衡，所以仍用深度学习的方式去获得尺度间的融合特征，并在高斯

分布柔化标签监督下，最终得到了更好的检测效果。

此外本文所提出的方法在 2D 检测和 AOS 表现如下：

表 2.7 2D 检测与 AOS 对比结果

方法	AP|2D			AOS		
	Easy	Mod	Hard	Easy	Mod	Hard
MonoDETR[73]	95.89	87.22	81.63	90.92	80.41	74.38
M3Detr(proposed)	**96.43**	**89.56**	**82.21**	**94.92**	**86.33**	**78.61**

从上表可以直观地看出 3D 目标检测的难度确实远高于 2D 目标检测，同时所提出的方法相比同级别的 3D 检测算法拥有更好的 2D 检测精度，并且在估计车辆朝向信息的 AOS 指标中也得到了更好的表现。综上所述，本文所提出的模型在实现更好的 2D 车辆检测任务下，同时提高了 3D 检测精度。

2.5 本章小结

本章具体介绍了所提出的单目 3D 车辆检测算法。首先给出了模型整体结构，包括从主干网络分出两个子网络分别学习深度和图像特征，以及采用深度引导的方式将两者特征融合。然后具体的说明所设计深度估计网络，及其转换为高斯柔和的分类问题，其最后的特征图使用 Transformer 进行相对深度学习并与深度估计值作为深度位置编码相结合完成深度特征编码，图像编码则采用 Deformable DETR 的方法，其核心为多尺度可变形注意力机制，然后在解码器中采用深度引导的方式将深度特征与图像特征进行融合，最后说明了不同任务对应的检测头及其损失函数。在数据集中进行验证与对比，可以看出所提出的方法具有更好的表现。

第 3 章 基于 DeepSORT 的多目标车辆跟踪

3.1 引言

在自动驾驶中，只有对同一车辆实现持续的跟踪，才能从其历史的动态状态中学习其运动特性和潜在意图等等。所以本章基于第二章的目标检测算法，建立在车辆视角下的多目标车辆跟踪模型。车辆跟踪问题是指在视角里任意时刻出现的同一车辆都能对应到唯一标识 ID，而在驾驶视角下又额外包含了自车运动的影响。前者主要有两个难点：如何匹配运动中的车辆，以及车辆在随时间的运动过程中会被遮挡，如何避免再出现后产生的标识变换（ID Switch）现象；而后者面临的难点是自车运动会使得对目标车辆运动估计变得不准确，匹配效率低。所以，本章基于 DeepSORT 框架，采用卡尔曼滤波算法对目标车辆位置进行预测，同时通过光流法计算两帧图像的仿射变换，对自车视角进行相机运动补偿，从而更好地估计与当前帧检测位置的距离；对于再次识别重新出现的车辆目标，通过设计外观特征网络来得到获取其的外观标识，进而估计与跟踪目标的外观相似程度；最后在匹配阶段综合考虑运动和外观两个方面来实现多目标车辆在帧间的关联匹配，从而实现多目标跟踪。

3.2 多目标跟踪问题描述

多目标跟踪是将多个关注的目标赋予独有的标识，如图 3.1 中的每个车辆独有 0，1，2 等 ID，并在视频或图像序列中持续的跟踪到该标识的目标，从而可以获取目标的相关时序信息。简单的跟踪如图 3.1 所示具体经历如下三种情况：

（1）产生。如在第一幅图片发现要跟踪的目标，赋予 0 到 4 的 ID，等到了第二幅图片又识别到了新的目标，但是考虑到可能是检测算法识别有误，所以本文则是判断是否在连续三帧中都能识别到该目标，才将其定义为跟踪目标，再赋予新的 ID：5，6；

（2）匹配。如在相邻的图中都能找到 ID 为 3 的车，并且确实为同一辆车则匹配成功，如果将同一辆车在其他图中赋予其它 ID，则称为 ID Switch。未匹配的情况具体可以又可以分为未匹配的跟踪（Unmatched Track）和未匹配的检测（Unmatched Detection）。ID 为 0 的车在前两张图中已赋予唯一 ID，并且都能跟踪到，但是在第三图中丢失，即该跟踪目标（Track）没有可以继续匹配的检测目标，称为未匹配的跟

踪。未匹配的检测是指识别到了目标，但没有可以匹配的已有的跟踪目标，此时就需要判断是否为要为跟踪的目标并生成新的 ID，还是误识别，即情况（1）。

（3）消亡。ID 为 0 的车在第三幅图中未出现，可能是目标检测算法未识别，也可能是被暂时遮挡，也可能是确实不存在了，所以在本文中会持续预测 40 帧，如果期间没有再次识别到，则将该跟踪目标定义为消亡，这个 ID 也就不会再次出现。

需要说明的是，图 3.1 为在移动的自车坐标系下观察周围的目标，如 ID 为 5 的车辆停在路边静止不动，此时在自车坐标系下，该车运动模型为反向的自车运动，而 ID 为 6 的车辆也沿着道路在运动，所以该车的运动模型为对应车辆运动和自车运动的叠加。

图 3.1 多目标跟踪示意图

3.3 多目标跟踪模型

本章基于 DeepSORT 框架实现多目标跟踪，如图 3.2 所示，其中蓝色模块为本文改进和添加的部分，该框架主要包含如图四个流程：观测、预测、匹配、更新：

（1）观测：将视频流中的每一帧图像输入到目标检测网络中，得到目标的 2D 边界框和 2D、3D 的位置信息，前者输入到外观特征网络学习得到该目标的表征特征，后者则是投影到像素坐标系下作为观测的位置信息；

（2）预测：对于已匹配的跟踪目标，在光流法的运动补偿下，根据卡尔曼滤波预测其在下一帧的位置，用以与后续观测的匹配；

（3）匹配：将已存在的跟踪目标和当前帧的检测目标进行数据关联的过程，相同目标赋予同一 ID，新目标则要作为新的跟踪目标赋予新 ID。匹配具体包含级联匹配和 GIoU 匹配两个过程，前者结合表征距离和位置距离进行第一次匹配，后者对未匹配的检测和跟踪进行 GIoU 的二次匹配；对于第二次匹配结果输入到产生和消亡的模块。

（4）更新：根据匹配结果，一方面位置信息结合预测和观测进行卡尔曼滤波更新，另一方面结合当前帧所观测的目标特征进行表征特征更新。

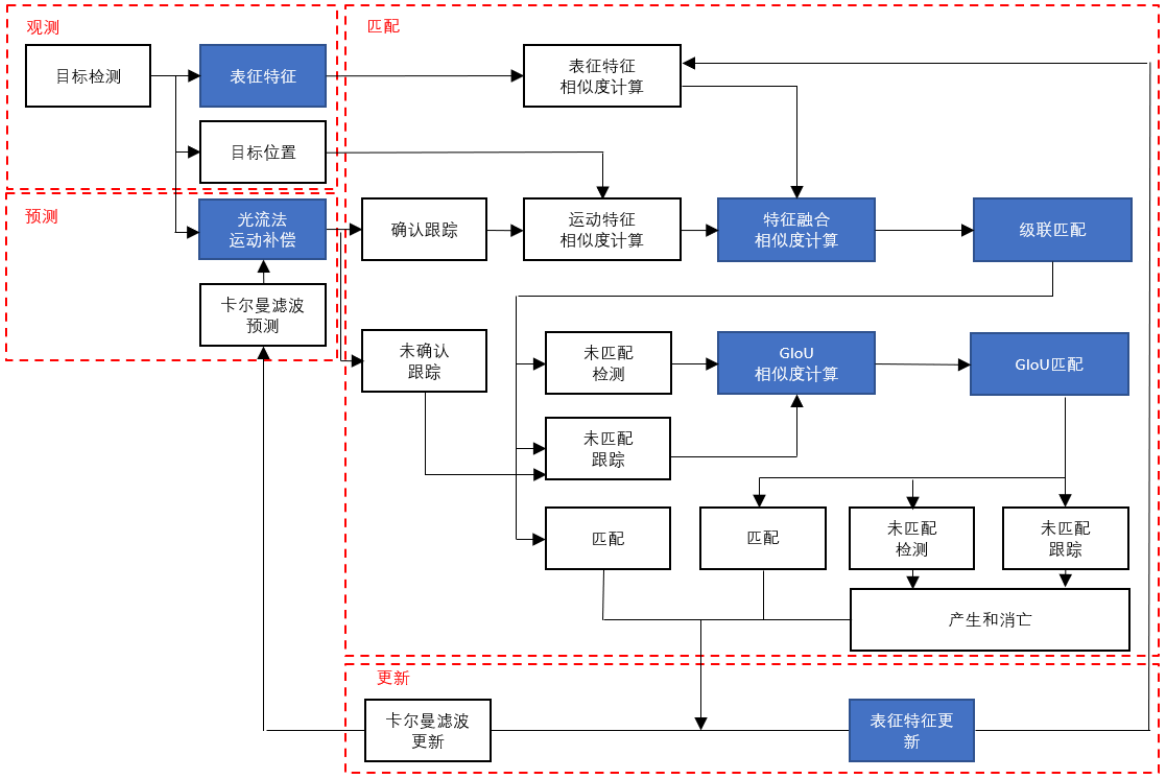


图 3.2 DeepSORT 跟踪框架

3.3.1 卡尔曼滤波运动模型

卡尔曼滤波算法的原理是利用卡尔曼增益来进行修正状态预测值，使其逼近真实值。假设离散线性动态系统模型如下：

$$\begin{cases} x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + \omega_{k-1} \\ y_k = H_k x_k + \upsilon_k \end{cases} \tag{3.1}$$

式中，k 对应离散时刻，x_k 为状态变量，y_k 为观测变量，u_k 为输入信号，$\omega_k \sim N(0, Q_k)$ 为服从高斯分布的过程激励噪声，$\upsilon_k \sim N(0, R_k)$ 为服从高斯分布的观测噪声，A_k 为状态转移矩阵，B_k 为控制输入矩阵，H_k 为状态观测矩阵。

上式中的真实值 x_{k-1} 并不能完全准确的得到，所以用状态的最优估计，也称后验估计的 $\overset{+}{x}_{k-1}$ 进行替代，进而求得在运动模型中下一时刻的预测状态，也称 k 时刻的先验状态估计 $\overset{-}{x}_k$：

$$\overset{-}{x}_k = A_{k-1}\overset{+}{x}_{k-1} + B_{k-1}u_k + \omega_{k-1} \tag{3.2}$$

状态估计的协方差也分为先验和后验：

$$\begin{aligned} P_k^+ &= E[(x_k - \hat{x}_k^+)(x_k - \hat{x}_k^+)^T] \\ P_k^- &= E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T] \end{aligned} \tag{3.3}$$

先验估计协方差公式如下：

$$\begin{aligned} P_k^- &= E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T] \\ &= E[(A_{k-1}x_{k-1} + B_{k-1}u_{k-1} - (A_{k-1}\hat{x}_{k-1}^+ + B_{k-1}u_{k-1} + \omega_{k-1})) \\ &\qquad (A_{k-1}x_{k-1} + B_{k-1}u_{k-1} - (A_{k-1}\hat{x}_{k-1}^+ + B_{k-1}u_{k-1} + \omega_{k-1}))^T] \\ &= E[(A_{k-1}(x_{k-1} - \hat{x}_{k-1}^+) - \omega_{k-1})(A_{k-1}(x_{k-1} - \hat{x}_{k-1}^+) - \omega_{k-1})^T] \\ &= E[A_{k-1}(x_{k-1} - \hat{x}_{k-1}^+)(x_{k-1} - \hat{x}_{k-1}^+)^T A_{k-1}{}^T + \omega_{k-1}\omega_{k-1}{}^T - \\ &\qquad A_{k-1}(x_{k-1} - \hat{x}_{k-1}^+)\omega_{k-1}{}^T - \omega_{k-1}(x_{k-1} - \hat{x}_{k-1}^+)^T A_{k-1}{}^T] \\ &= A_{k-1}E[(x_{k-1} - \hat{x}_{k-1}^+)(x_{k-1} - \hat{x}_{k-1}^+)^T]A_{k-1}{}^T + Q_{k-1} \\ &= A_{k-1}P_{k-1}^+ A_{k-1}{}^T + Q_{k-1} \end{aligned} \tag{3.4}$$

卡尔曼滤波作为线性预测，是通过衡量实际观测与先验估计观测的误差，来对对推理的先验估计进行线性补偿，从而得到更新的后验估计，有：

$$\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - H_k\hat{x}_k^-) \tag{3.5}$$

其中 K_k 为线性卡尔曼增益。更新后实际状态 x_k 与后验估计 $\hat{x}_k^+$ 的误差表示为：

$$\begin{aligned} E[\varepsilon_{x,k}^+] &= E[x_k - \hat{x}_k^+] \\ &= E[x_k - (\hat{x}_k^- + K_k(y_k - H_k\hat{x}_k^-))] \\ &= E[\varepsilon_{x,k}^- - K_k(H_k x_k + \nu_k - H_k\hat{x}_k^-)] \\ &= E[\varepsilon_{x,k}^- - K_k(H_k(x_k - \hat{x}_k^-) + \nu_k)] \\ &= E[(I - K_k H_k)\varepsilon_{x,k}^- - K_k\nu_k] \\ &= (I - K_k H_k)E[\varepsilon_{x,k}^-] - K_k E[\nu_k] \end{aligned} \tag{3.6}$$

所以后验估计的协方差有：

$$\begin{aligned} P_k^+ &= E(\varepsilon_{x,k}^+ \varepsilon_{x,k}^{+T}) \\ &= E[((I-K_kH_k)E[\varepsilon_{x,k}^-]-K_kE[\nu_k])((I-K_kH_k)E[\varepsilon_{x,k}^-]-K_kE[\nu_k])^T] \\ &= (I-K_kH_k)E(\varepsilon_{x,k}^- \varepsilon_{x,k}^{-T})(I-K_kH_k)^T + K_kR_kK_k^T \end{aligned} \tag{3.7}$$

为使实际状态与后验估计尽可能地接近，期望误差最小，所以对 P_k^+ 其求增益 K_k 偏导，并令其为 0，有：

$$\begin{aligned} &\frac{\partial P_k^+}{\partial K_k} = 2(I-K_kH_k)P_k^-(-H_k)^T + 2K_kR_k = 0 \\ &K_k = P_k^-H_k^T(R_k + H_kP_k^-H_k^T)^{-1} \end{aligned} \tag{3.8}$$

将(3.8)带入(3.7)得到了协方差的更新公式：

$$P_k^+ = (I-K_kH_k)P_k^- \tag{3.9}$$

从而得到卡尔曼滤波的预测方程(3.2)和(3.4)，和更新方程(3.8),(3.5),(3.9)：

$$\begin{cases} \hat{x}_k^- = A_{k-1}\hat{x}_{k-1}^+ + B_{k-1}u_k \\ P_k^- = A_{k-1}P_{k-1}^+A_{k-1}^T + Q_{k-1} \end{cases} \tag{3.10}$$

$$\begin{cases} K_k = P_k^-H_k^T(R_k + H_kP_k^-H_k^T)^{-1} \\ \hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - H_k\hat{x}_k^-) \\ P_k^+ = (I-K_kH_k)P_k^- \end{cases} \tag{3.11}$$

本章使用上述卡尔曼滤波对目标的位置进行运动预测，通过第二章目标检测的结果分析可以看出，2D 目标检测精度远高于 3D 目标检测，所以设计输入输出的状态为 2D 检测相关信息：

$$\begin{cases} X = \left[x, y, a, h, \dot{x}, \dot{y}, \dot{a}, \dot{h}\right]^T \\ Y = \left[x, y, a, h\right]^T \end{cases} \tag{3.12}$$

其中，(x, y) 为目标中心点像素坐标，a 为 2D 边界框的宽高比，h 为边界框的高，后四项为前四项的变化率。

不考虑控制输入，并假设目标匀速运动有：

$$\begin{cases} x_{t+1} = x_t + \dot{x}_t\Delta t \\ y_{t+1} = y_t + \dot{y}_t\Delta t \\ a_{t+1} = a_t + \dot{a}_t\Delta t \\ h_{t+1} = h_t + \dot{h}_t\Delta t \end{cases} \tag{3.13}$$

记 X 前四项为 X_{pos}，后四项为 X_{vel}，则将(3.13)表示为如下空间表达式：

$$\begin{cases} \begin{bmatrix} X_{pos} \\ X_{vel} \end{bmatrix}_{t+1} = \begin{bmatrix} \mathbf{1} & \Delta t \\ 0 & \mathbf{1} \end{bmatrix} \begin{bmatrix} X_{pos} \\ X_{vel} \end{bmatrix}_{t} \\ Y = \begin{bmatrix} \mathbf{1} & 0 \end{bmatrix} X \end{cases} \tag{3.14}$$

3.3.2 光流法相机运动补偿

由于自车的运动会使得卡尔曼滤波对目标车辆的运动预测总会有扰动而变得不准确，所以这里使用光流法计算相邻帧的图像因为相机运动产生仿射变换，来作为前者的运动补偿。

光流法是描述像素随时间在图像之间运动的方法，如果只计算部分像素的运动称为稀疏光流，比如 Lucas-Kanade 光流[75]，计算所有像素则为稠密光流，考虑后者是对图像进行逐点匹配，计算所有点的偏移量后得到光流场，再进行配准，计算量过大，所以本文采用 LK 算法。为提取图片中具有代表性的部分像素，首先采用 FAST 提取特征点。

FAST 算法先从已转换为灰度图的图片中选取一个坐标点，其灰度值为I_p，然后得到以该点为圆心，半径为 3 的 Bresenhanm 圆，如图 3.3 所示，该圆上共有 16 个离散点，设定灰度阈值$t(=100)$，如果有$n(=9)$个连续的像素点的灰度值都比圆心I_p+t大或者比I_p-t小，则认为该圆心坐标点是角点。由于对比 16 个点效率太低，所以实际只检测 1，5，9，13 这 4 个位置，如果 1，9 位置的灰度值满足条件再对比 5，13 两个位置，当至少三个点满足阈值条件才能将该圆心称为角点。角点附近同样会有多个类似的角点，所以还需要进行非极大抑制处理，从而得到最后的特征点。

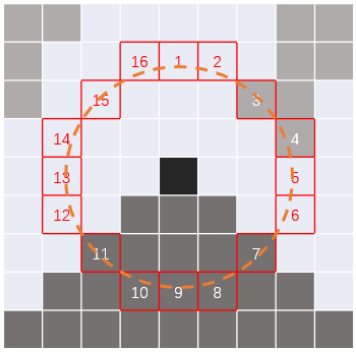


图 3.3 FAST 示意图

计算特征点的运动是建立在光流法基本假设上：灰度不变假设，即同一个空间

点的像素灰度值，在连续的图像中保持不变，如图 3.4 所示。

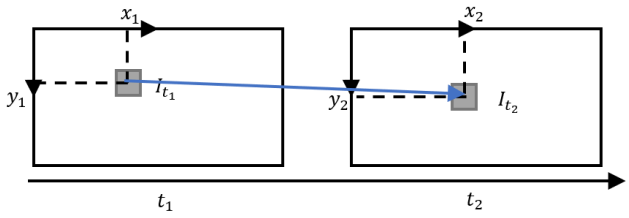


图 3.4 灰度不变假设

在 LK 光流中，图像可以看作是时间的函数，而在t时刻，位于(x,y)位置的像素的灰度值写作$I(x,y,t)$。该像素点在$t+dt$时刻运动到$(x+dt,y+dt)$位置，又因为灰度值不变假设，从而有：

$$I(x+dx,y+dy,t+dt)=I(x,y,t) \tag{3.15}$$

对左边进行泰勒展开并只保留一阶项有：

$$I(x+dx,y+dy,t+dt)\approx I(x,y,t)+\frac{\partial I}{\partial x}dx+\frac{\partial I}{\partial y}dy+\frac{\partial I}{\partial t}dt \tag{3.16}$$

将(3.16)带回(3.15)有：

$$\begin{aligned}&\frac{\partial I}{\partial x}dx+\frac{\partial I}{\partial y}dy+\frac{\partial I}{\partial t}dt=0\\ \Leftrightarrow\ &\frac{\partial I}{\partial x}\frac{dx}{dt}+\frac{\partial I}{\partial y}\frac{dy}{dt}=-\frac{\partial I}{\partial t}\end{aligned} \tag{3.17}$$

其中dx/dt，dy/dt分别表示像素在x,y方向上的运动速度，记为u,v。同时$\partial I/\partial x,\partial I/\partial y$表示图像的灰度值在$x,y$的方向梯度，分别记为$I_x,I_y$。把图像灰度值对时间的变化量记为$I_t$，所以(3.17)的矩阵形式为：

$$\begin{bmatrix}I_x & I_y\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}=-I_t \tag{3.18}$$

在 LK 光流中假设某一窗口内的像素具有相同的运动，而窗口内所关注的特征角点为w数量的像素点，因而共有w个方程：

$$\begin{bmatrix}I_x & I_y\end{bmatrix}_k\begin{bmatrix}u\\ v\end{bmatrix}=-I_{tk},\quad k=1,\ldots,w \tag{3.19}$$

因为是关于u,v的超定线性方程，可以用最小二乘法求解：

$$\begin{bmatrix} u \\ v \end{bmatrix}^{*} = -(A^{T}A)^{-1}A^{T}b, \quad A = \begin{bmatrix} \begin{bmatrix} I_x & I_y \end{bmatrix}_1 \\ \vdots \\ \begin{bmatrix} I_x & I_y \end{bmatrix}_k \end{bmatrix}, b = \begin{bmatrix} I_{t1} \\ \vdots \\ I_{tk} \end{bmatrix} \tag{3.20}$$

上述为单层光流法，对于多层光流法是对同一图像进行缩放，得到不同分辨率的图像，先从顶层开始计算，然后把上一层的追踪结果作为下一层光流的初始值，从而形成金字塔光流法。

3.3.3 外观特征网络

目标在因被遮挡等问题消失后重新出现，就需要再次匹配到原来对应的目标，这个 ReID 的过程可以通过衡量该目标和已存储的其它多目标的表征特征之间的相似度来实现。

原外观特征提取网络主要通过卷积网络和残差结构的组合来实现，表中$[\bullet]$表示残差单元，同 2.3.1 类似，是将 Layer 的输入与输出相加后，再经过激活函数 ReLU 得到该层的最后结果，具体设计如下表：

表 3.1 外观特征提取网络

网络层	输出维度/(C, H, W)	卷积核/输出维度/步长
Input	(3, 64, 128)	-
Conv	(64, 64, 128)	3×3, 64, stride=1
MaxPool	(64, 32, 64)	3×3,max pool, stride=2
Layer1	(64, 32, 64)	$\begin{bmatrix} 3\times3,64,s=1 \\ batchnorm+relu \\ 3\times3,64,s=1 \end{bmatrix}\times2$
Layer2	(128, 16, 32)	$\begin{bmatrix} 3\times3,128,s=2 \\ batchnorm+relu \\ 3\times3,128,s=1 \end{bmatrix}\times2$
Layer3	(256, 8, 16)	$\begin{bmatrix} 3\times3,256,s=2 \\ batchnorm+relu \\ 3\times3,256,s=1 \end{bmatrix}\times2$
Layer4	(512, 4, 8)	$\begin{bmatrix} 3\times3,512,s=2 \\ batchnorm+relu \\ 3\times3,512,s=1 \end{bmatrix}\times2$
AvgPool	(512, 1)	4×8, avg pool, stride=1 reshape(512,1)
Linear	(128, 1)	

为了提高识别准确率，本文在此基础上基于 PAN[76]结构进行多尺度特征融合的改进。不同于 FPN 是自顶向下的，将高层特征通过上采样和低层特征不断融合得到

更新的特征图，PAN 则是在 FPN 层的后面还添加了一个自底向上的特征金字塔，如图 3.5 所示，图中 Layer’为上采样卷积，先经过双线性插值到同一尺寸后，在经过卷积到同一通道数，Layer’’为卷积下采样，通过设定步长实现下采样。

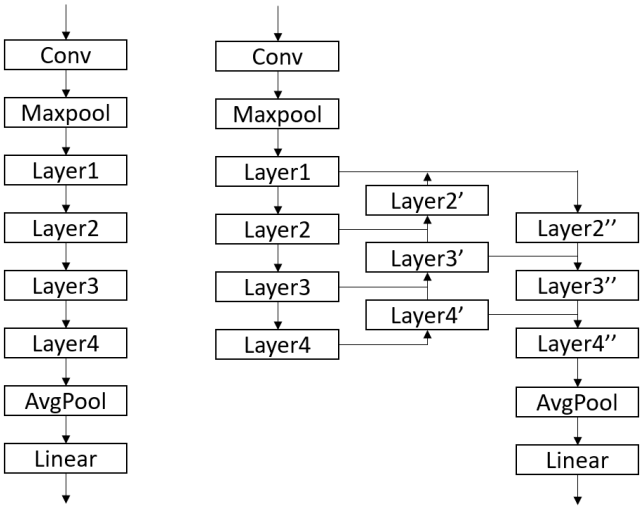


图 3.5 原卷积残差网络（左），基于 PAN 的多尺度融合结构（右）

针对多车辆目标的跟踪，外观特征训练使用车辆重识别数据集 VeRi-776[77]，如图 3.6 所示。该数据集包含 776 辆车共计 50000 多张图像，由 20 台摄像机在一平方公里的北京城市区域经由 24 小时拍摄得到，具体每辆车都会被 2~18 台相机抓拍从而获得其在不同视角，照明，分辨率，遮挡情况的图像。

图 3.6 VeRi-776 数据示例

对于多分类任务，本节采用了简单的 Top-1 正确率指标进行评价，该指标表示的是预测正确的类别的样本数占总样本数的概率。训练如图 3.7 所示，共对比了四种特

征融合的结构，Ori 表示的是原残差卷积网络，FPN 是对最后三层进行自顶向下融合，Dense 是对原残差卷积进行简单的再次跨层稠密化连接，PAN 如图 3.5 右所示，包括自顶向下和自底向上的融合。

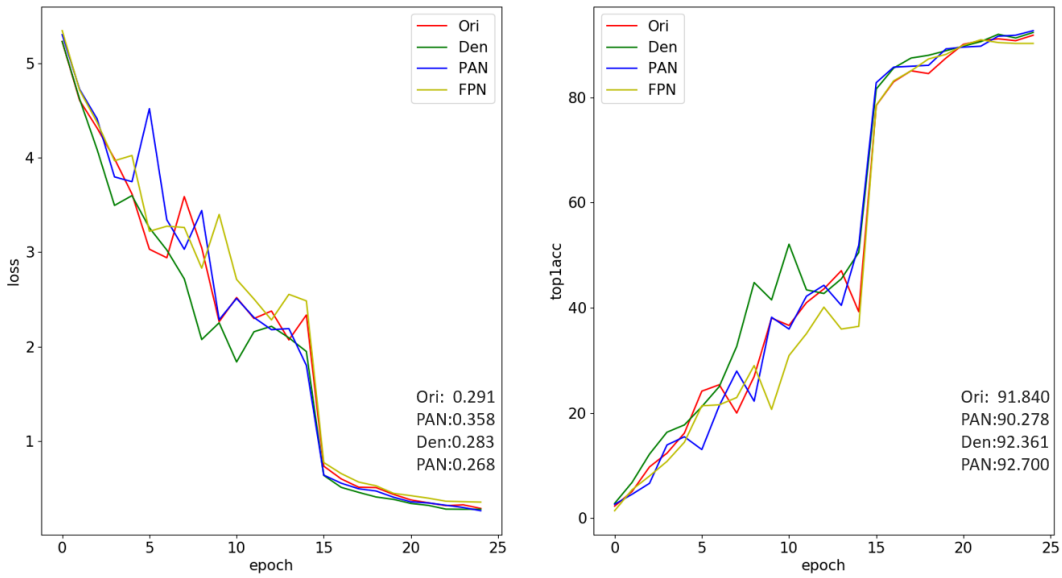


图 3.7 训练损失及 Top1 正确率

可以看出基于 PAN 结构的多尺度融合比原网络正确率提高了 0.76%，因此本文采用的改进后的外观特征网络来对表征特征进行提取。

3.3.4 数据关联

数据关联是指对于每一帧检测到的目标和上一帧甚至更早的跟踪目标进行匹配，并尽可能地保证关联的为同一车辆，并赋予同一 ID。匹配主要依据目标之间相似度，包括运动相似度和表征特征相似度，所构成的不同成本矩阵进行衡量。同时为了尽可能的完成帧间关联，共进行了两次匹配，第一次为表征与运动联合的级联匹配，第二次仅为 GIoU 匹配。

（1）级联匹配

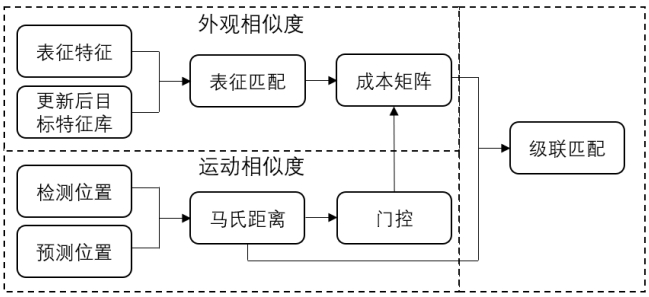


图 3.8 级联匹配结构图

在第t帧，通过目标检测可以识别到第i个目标车辆对应的2D边界框，将其所对应的目标图像$\mathrm{bbox}_{t,i}$输入到外观特征网络提取对应的表征特征$f_{t,i}$：

$$f_{t,i} = reid(\mathrm{bbox}_{t,i}) \tag{3.21}$$

实际上在2D检测中，对于遮挡的目标进行表征特征提取会参杂前景的外观信息，所以如果当前第i个目标特征$f_{t,i}$直接与上一帧的候选的第j个目标特征库$e_{t-1,j}$进行匹配时，会因为不够相似造成目标丢失，另一方面特征库是由新检测的特征组合而成，所包含的特征是按时序分离的，为了增加特征在帧间的连贯性，特征库使用如下方法与每帧已匹配的特征进行更新：

$$e_{t-1,j} = (1-\lambda)e_{t-2,j} + \lambda f_{t-1,j} \tag{3.22}$$

其中λ表示特征分配权重，在本文设计为0.1。

对外观相似度的匹配便是f_i^t的集合与e_j^{t-1}的集合间的表征匹配，表征匹配也分为两个步骤：求余弦距离成本矩阵和运动特征相似度。余弦相似度表示的是两个特征向量间的夹角余弦值，余弦距离则是用单位值减去余弦相似度：

$$D_{\cos}(i,j) = \min_k (1 - \frac{f_{t,i} e_{t-1,j}^{(k)}}{\left\| f_{t,i} \right\| \left\| e_{t-1,j}^{(k)} \right\|} \mid k \in R_{t-1,j}) \tag{3.23}$$

式中$e_{t-1,j}^{(k)}$表示的是在$t-1$时刻更新的第j个跟踪目标对应的第k个表征特征，$R_{t-1,j}$为依据时序保留的最近40帧图像的表征特征。

运动特征相似度的计算如图3.8中的左下框，是指当跟踪的目标符合假定的运动模型时，其在当前帧所检测的位置与经过上一帧预测的位置在空间的近似程度，因为马氏距离相比于欧式距离修正了在不同维度下尺度不一致的问题，不受量纲影响，所以采用该方法进行衡量：

$$D_{mv}(i,j) = (d_i - p_j)^T \Sigma_j^{-1} (d_i - p_j) \tag{3.24}$$

式中d_i表示检测到的第i个边界框位置，p_j表示预测的第j个已跟踪目标的边界框位置，Σ_j表示卡尔曼滤波中的协方差矩阵。

运动特征相似度可以通过设定阈值来筛选表征特征匹配程度的余弦距离成本矩阵，用来防止目标匹配到空间距离过远的车辆。阈值设定为χ^2分布自由度为4的95%分位点$D_{threshold} = 9.4877$。

$$D_{\cos} = D_{\cos}[D_{mv} < D_{threshold}] \tag{3.25}$$

综合运动相似度提供了目标经过短期预测在空间位置的信息，外观相似度提供

了目标在长期运动中表征特征的信息。所以将两者进行线性加权得到最终级联匹配的相似度矩阵：

$$D = \lambda D_{\cos} + (1-\lambda) D_{mv} \tag{3.26}$$

式中λ为表征特征与运动特征的分配权重，本文为 0.98。

（2）GIoU 匹配

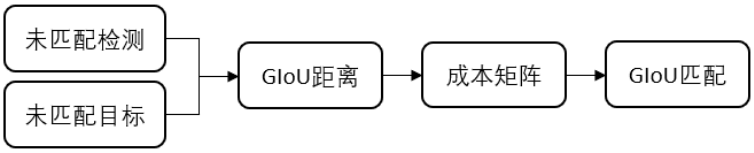


图 3. 9 GIoU 匹配结构图

级联匹配是在限定运动范围内的，以外观相似度为主的级联匹配。经过第一次匹配筛选后，为了尽可能匹配检测与跟踪目标，所以再将未匹配的检测和跟踪目标，通过衡量 GIoU 值再进行一次以运动相似度为主的匹配，GIoU 距离类似公式(2.17)，转写如下：

$$GIoU(i,j) = 1 - \left(\frac{\left|S_{p,j} \cap S_{d,i}\right|}{\left|S_{p,j} \cup S_{d,i}\right|} - \frac{\left|A_{ij} - S_{p,i} \cup S_{d,i}\right|}{\left|A_{ij}\right|}\right) \tag{3.27}$$

式中$S_{p,j}$表示的是未匹配的第j个跟踪目标其预测的 2D 边界框所占的面积，$S_{d,i}$表示的是未匹配的第i个检测目标的 2D 边界框所占的面积，A_{ij}表示两边界框的最小闭包区面积。

对于上述两次匹配所构建的相似度成本矩阵，都采用匈牙利算法完成配对。

（3）匈牙利算法

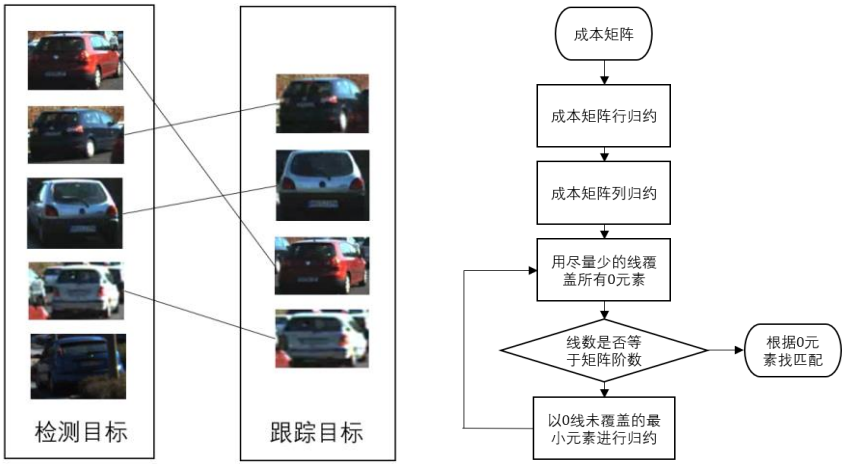


图 3.10 二分匹配（左），匈牙利算法（右）

匈牙利算法(Hungarian Algorithnm)可以用来解决二分图匹配的问题，如图 3.10 左所示，二分图在本文中具体表示为两个集合，一个是当前帧的检测集合，另一个是前一帧中各跟踪目标所对应的候选集合。其具体流程图如图 3.10 右所示，行归约是指成本矩阵每一行减去改行的最小元素值，列归约是在列中进行，用尽量少的线覆盖所有 0 元素即找 0 线的过程，就是从最小成本开始匹配，如果线数等于矩阵最小阶数，即找到了当前的最大匹配，否则需要从“暂未匹配”的未被 0 线覆盖的区域中找到最小成本元素，并减去，如此循环直到完成最大匹配。

3.4 实验结果与分析

3.4.1 实验数据集

本章采用 KITTI 的跟踪数据集，由 21 个训练序列和 29 个测试序列组成。因为只有训练序列带有标签，且本文所搭建的多目标跟踪模型并未使用过训练集中的部分进行训练，所以将其中以车辆为主的序列作为测试集进行验证。

3.4.2 评价指标

（1）多目标跟踪准确率 MOTA（Multi Object Tracking Accuracy）

MOTA 指标衡量的是多目标跟踪算法精确性，该值越大跟踪效果越好，公式如下：

$$MOTA = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t GT_t} \tag{3.28}$$

其中 FN 表示漏检率，是未检测的真实目标数量，FP 表示误检率，是将其它物体错误判定为目标的数量；$IDSW$ 表示 ID 切换次数；GT 表示真实边界框数量。

（2）多目标跟踪精度 MOTP（Multiple Object Tracking Precision）

MOTP 衡量的是多目标的定位精度，非跟踪器性能，该值越大越好：

$$MOTP = \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t} \tag{3.29}$$

其中 d 为检测目标 i 与分配的真值之间在所有帧的平均度量距离，用边界框的重叠率（Overlap Rate）表示，c 为当前帧匹配成功的数目。

（3）IDF1

IDF1 指标代表被检测和跟踪的目标中获取正确 ID 的检测目标比例，综合考虑

了 ID 准确率（ID Precision）和 ID 召回率（ID Recall），是两者的调和均值。

$$IDP = \frac{IDTP}{IDTP + IDFP} \tag{3.30}$$

$$IDR = \frac{IDTP}{IDTP + IDFN} \tag{3.31}$$

$$IDF1 = \frac{2IDTP}{2IDTP + IDFP + IDFN} \tag{3.32}$$

式中 IDP 表示的是每个车辆框中车辆 ID 识别精确度，IDTP 和 IDFP 分别代表真正 ID 数和假正 ID 数，IDR 表示 ID 识别召回率，IDFN 为假负 ID 数。

（4）主要跟踪 MT（Mostly Tracked）

MT 表示成功跟踪的时长超过 80%的目标的数量，该值越大越好。

（5）主要丢失 ML（Mostly Lost）

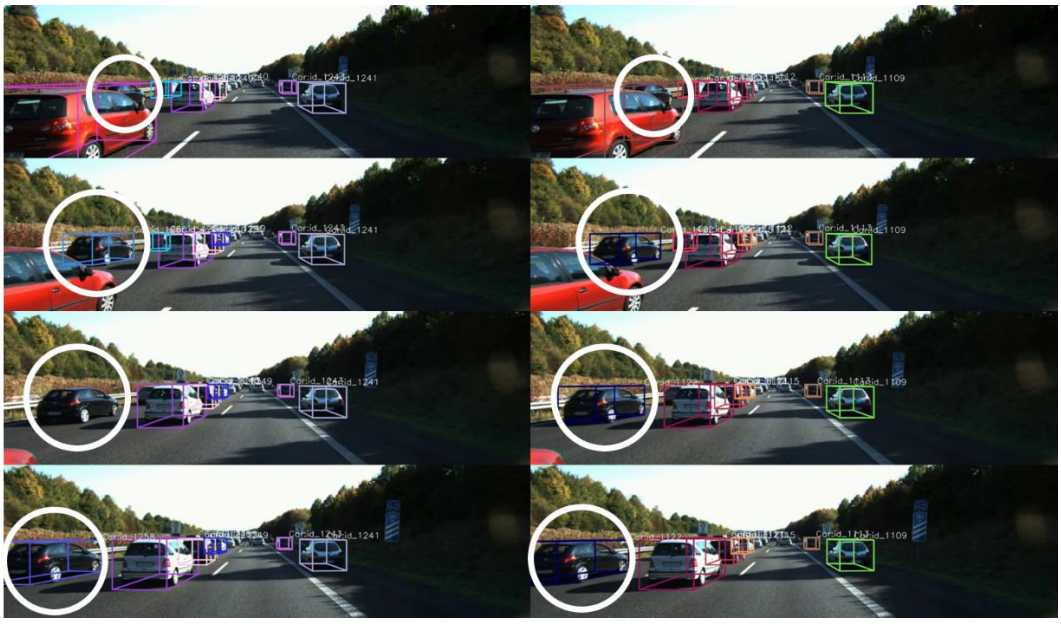
ML 表示成功跟踪时长小于 20%的目标的数量，该值越小越好。

（6）ID 切换 IDSW（ID Swich）

表示正在跟踪的目标在过程中 ID 切换次数，该值越小越好。

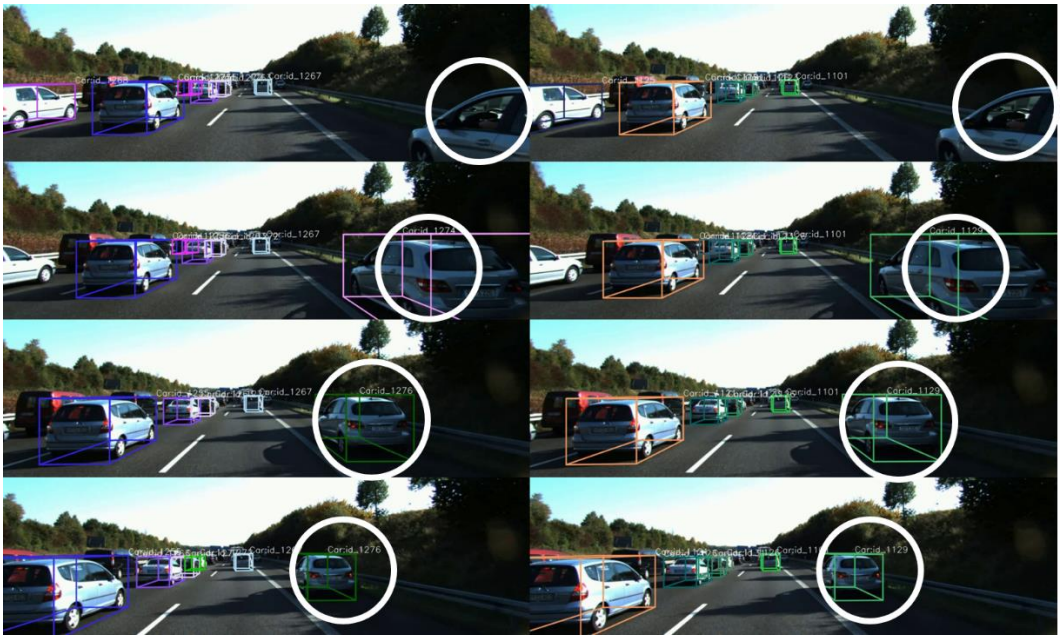
3.4.3 实验结果与对比

（1）实验结果可视化



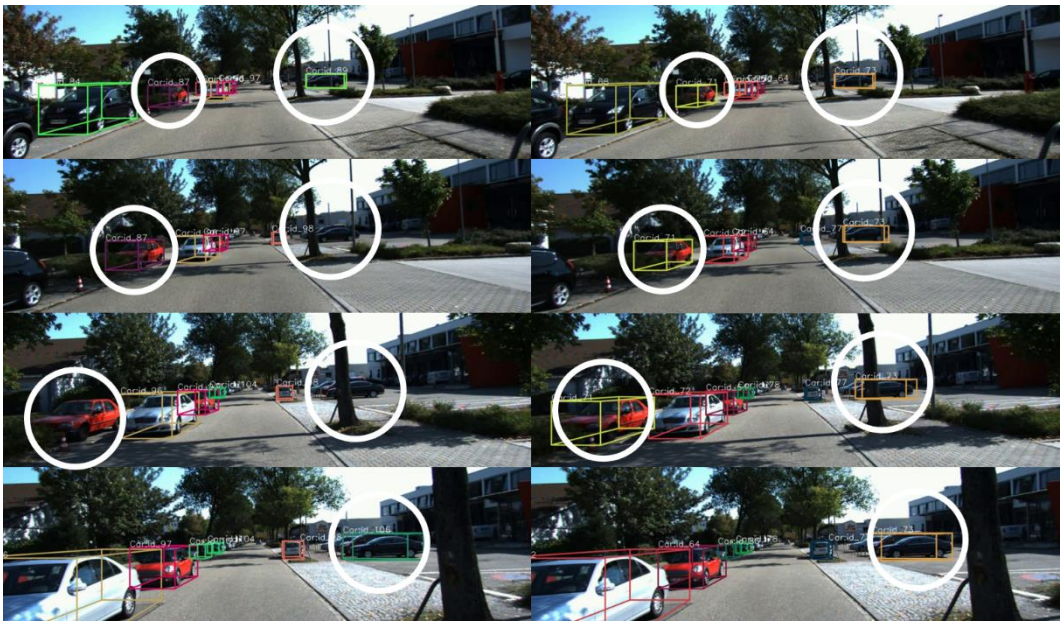
（a）DeepSORT　　(b) 本文改进后

图 3.11 多目标跟踪可视化一（每隔 5 帧）



（a）DeepSORT　　(b) 本文改进后

图 3.12 多目标跟踪可视化二（每隔 5 帧）



（a）DeepSORT　　(b) 本文改进后

图 3.13 多目标跟踪可视化三（每隔 5 帧）

（a）DeepSORT　　(b)本文改进后

图 3.14 多目标跟踪可视化四（每隔 5 帧）

多目标跟踪可视化如图 3.11-3.14 中所示，在第二章提出的单目 3D 检测算法基础上，左图（a）为原 DeepSORT 生成的多目标跟踪结果，右图（b）为改进后的多目标跟踪结果。从图中可以看出，大部分目标都有较好的识别与跟踪，但在对比图中白色圆圈标注的车辆，原算法中则出现了跟踪丢失以及 ID 变换的情况。

在图 3.11（a）中对于原 DeepSORT 算法对于识别到的物体并没能通过卡尔曼滤波算法中进行很好的预测，从而将其识别为新的跟踪目标，导致 ID 改变，由 ID1256 变为 1258，而在经过光流法相机运动补偿之后，如图 3.11（b）所示，对于视角中识别到的车辆能够保持良好的跟踪效果，ID 一直为 1122。该现象也同样发生在图 3.12 中右侧超车的白车，原算法在跟踪时 ID 由 1274 变为 1276，改进后保持为同一 ID1129。在图 3.13 中左侧的红色车辆，在第三幅图（a）中突然丢失，这是因为原算法认为在当前帧识别的该车辆并未与前一帧相匹配，是新的车辆，但同时又还没有达到产生为新跟踪目标的连续帧数条件，所以丢失了 3D 框标注。除此之外，该图右侧的目标被前方树所遮挡便丢失了跟踪，再识别时原算法也将其认为是新目标，ID 由 89 变为 109，但在改进后的算法中则能够保持持续跟踪，说明外观特征及其连续性也有利于提升跟踪效果。在图 3.14 中则是对同一目标，连续识别错误，赋予了 3 个新的 ID，597，599，602，由此可见原 DeepSORT 算法在多目标跟中还有很多不足，对应的右图（b）为改进后的模型则在一定程度上解决了这样的问题。

（2）实验结果对比

表 3.2 多目标跟踪算法评价对比

方法	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	IDSW↓
DeepSORT	65.542	78.355	69.825	156	67	349
DeepSORT 改进后	**66.785**	**78.357**	**74.880**	**172**	**64**	**281**

上表展示了在具体评价指标下的对比结果，其中，↑表示该值越高越好，↓表示该值越低越好，可以看出改进后的算法在所有指标上都有所提高。MOTA 结果显示出改进的算法在多目标跟踪准确率提高了 1.243%，MOTP 提升不大是因为用的同一目标检测算法，而 IDF1 提升了 5.055%，能较明显的说明算法改进后能更加正确的跟踪目标，ID 改变数量也有明显的减少。综合上述对比结果，改进后的算法可以实现更稳定有效的多目标跟踪。

3.5 本章小结

为提升在驾驶视角下多目标跟踪的效果，本章基于 DeepSORT 的多目标跟踪算法上进行改进。首先根据第二章的车辆目标检测算法，采用识别准确率更高的 2D 检测结果作为车辆的运动状态，为了弥补自车运动产生的目标偏移，在基于卡尔曼滤波的方法上采用光流法进行运动补偿。在外观特征提取中，使用 PAN 结构进行多尺度特征融合，从而提高重识别准确率，并在 VeRi-776 数据集上进行了验证。为了更好的进行数据关联，在级联匹配的方式中融合了外观特征和运动特征，在第二次匹配中采用 GIoU 的方式，来衡量帧间目标的距离。此外，对于匹配的跟踪目标，其外观特征也进行了时序关联来更新，防止因遮挡造成的特征干扰。最后，在 KITTI 的跟踪数据集上进行可视化展示和指标验证对比，结果显示改进后的算法更加稳定有效。

第 4 章 考虑多目标点的车辆轨迹预测

4.1 引言

作为感知层下游，需要对周围环境的未来状态有个合适的预估，即对交通参与者未来的轨迹进行预测，否则突然的转向或者换道会对后续的决策规划与控制带来极大挑战。此外，实际交通环境中大多车辆都会遵守交通规则，在规定的车道区域内行驶，所以也应该将环境约束进行综合考虑。

本章在假设已知车辆历史轨迹与局部地图的基础上，提出考虑多目标点的车辆轨迹预测模型 TGCN（Target considered Graph Convolution Network），最主要的特点是将目标点（包括中点和终点）的环境特征融入到轨迹预测里。具体而言，该模型从车辆的历史时序状态，与车道的环境信息出发，分别提取车辆运动特征，及所处的环境特征。为实现先多特征融合，首先将环境特征约束赋予每个参与者，然后考虑到车车之间的互相影响，所以将交互信息进行传递。然后为了减少预测轨迹与实际真值的误差，先对中点进行预测，并将该点附近的环境特征赋予车辆。在实际环境中，尤其是在路口处，对车辆直行还是转向需要进行合理的判断，也就由此产生了多模态预测问题，本章进一步的预测多个终点，取其中置信度最高的终点，并将该点处的环境特征也融入到车辆特征内，其余终点则是在大量数据训练下，去自动学习分类成在不同环境约束下的多终点。至此轨迹预测问题就被重塑为已知起点，长时预测的中点和终点如何回归未来轨迹的问题。最后，将车辆信息，约束信息，车车交互信息，起点和目标点环境信息融合的特征进行解码，输出预测的未来轨迹。

4.2 预测问题描述

轨迹预测作为感知与决策规划之间承上启下的模块，是指在自车行驶过程中对目标车辆（Target Vehicle, TV）甚至周围的所有车辆（Surrounding Vehicle, SV）的轨迹进行预测。在实际情况中，周边的车辆一定程度上会与目标车辆产生交互，进而影响其未来的行为，比如前方车辆刹车，目标车辆为避免碰撞也会刹车等，并且这种影响会在局部范围内逐渐传播。因此，预测问题的第一部分输入为所有车辆的历史轨迹：

$$X = [h_{t-t_{his}}, h_{t-t_{his}+1}..., h_{t-1}, h_t] \tag{4.1}$$

其中h_t表示在t时刻的输入，具体包括目标车辆 T 和其他所有车辆 S 信息：

$$h_t = [T_t, S_t^i], i = 1,2,3,...,n \tag{4.2}$$

式中i表示第i辆车，T 与 S 的车辆信息输入相同，都为相邻时刻的相对距离，平均速度，和由其正余弦表示的方向信息，以及表示该轨迹点是否存在的标志位：

$$S_t^i = [dx_t^i, dy_t^i, v_t^i, \sin(\theta_t^i), \cos(\theta_t^i), flag_t^i] \tag{4.3}$$

预测的轨迹输出表示为：

$$Y = [P_0, P_i], i = 1,2,3,...,n \tag{4.4}$$

其中P_0表示目标车辆，P_i表示为周围第i辆车的预测轨迹点，表示为：

$$P_i = [[x_{t+1}^i, y_{t+1}^i], ..., [x_{t+t_p-1}^i, y_{t+t_p-1}^i], [x_{t+t_p}^i, y_{t+t_p}^i]] \tag{4.5}$$

其中t_p为预测时域。

预测问题的第二部分输入是环境信息，本文主要考虑车道中心线之间的空间关系。具体而言，车道中心线由 Argoverse[54]高精矢量地图获取得到，每一段中心线包含 10 个具体的路点，又因为两个路点才能组成一个线段，所以每一线段的具体位置由对应的两端点的中点$(v_i^{end} - v_i^{start})/2$表示，其对应的形状信息，包括长度和方向则由对应的两端点构成的矢量$(v_i^{end} - v_i^{start})$进行表示，从而每一段车道中心线由 9 个线段节点组成，如图所示，虚线代表车道线，实线代表车道中心线，每一段中心线由 9 个同色圆点表示，其中蓝色，绿色，橙色为非路口中直行，红色，黄色，白色表示在路口中直行，灰色表示右转。

明确车道的表示方法后，车道的空间关系是指每个路点之间的相对位置，比如橙色段车道中心线的路点 9，其前方衔接的是灰色路点 1 和白色路点 1，后方为橙色路点 8，左边是绿色路点 9，右边没有则记为 0。通过建立如上联系，从而可以构建其关于路点的拓扑结构图。

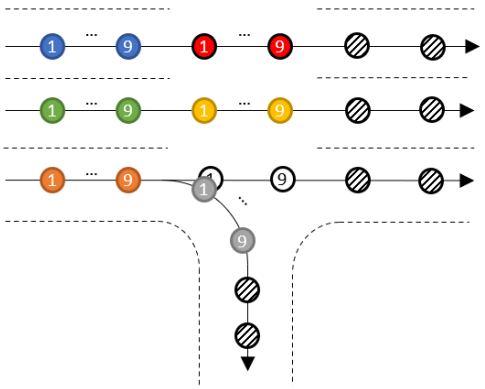


图 4.1 车道地图的拓扑结构

4.3 轨迹预测模型

提出的 TGCN 轨迹预测模型如图 4.2 所示，分为五个部分：

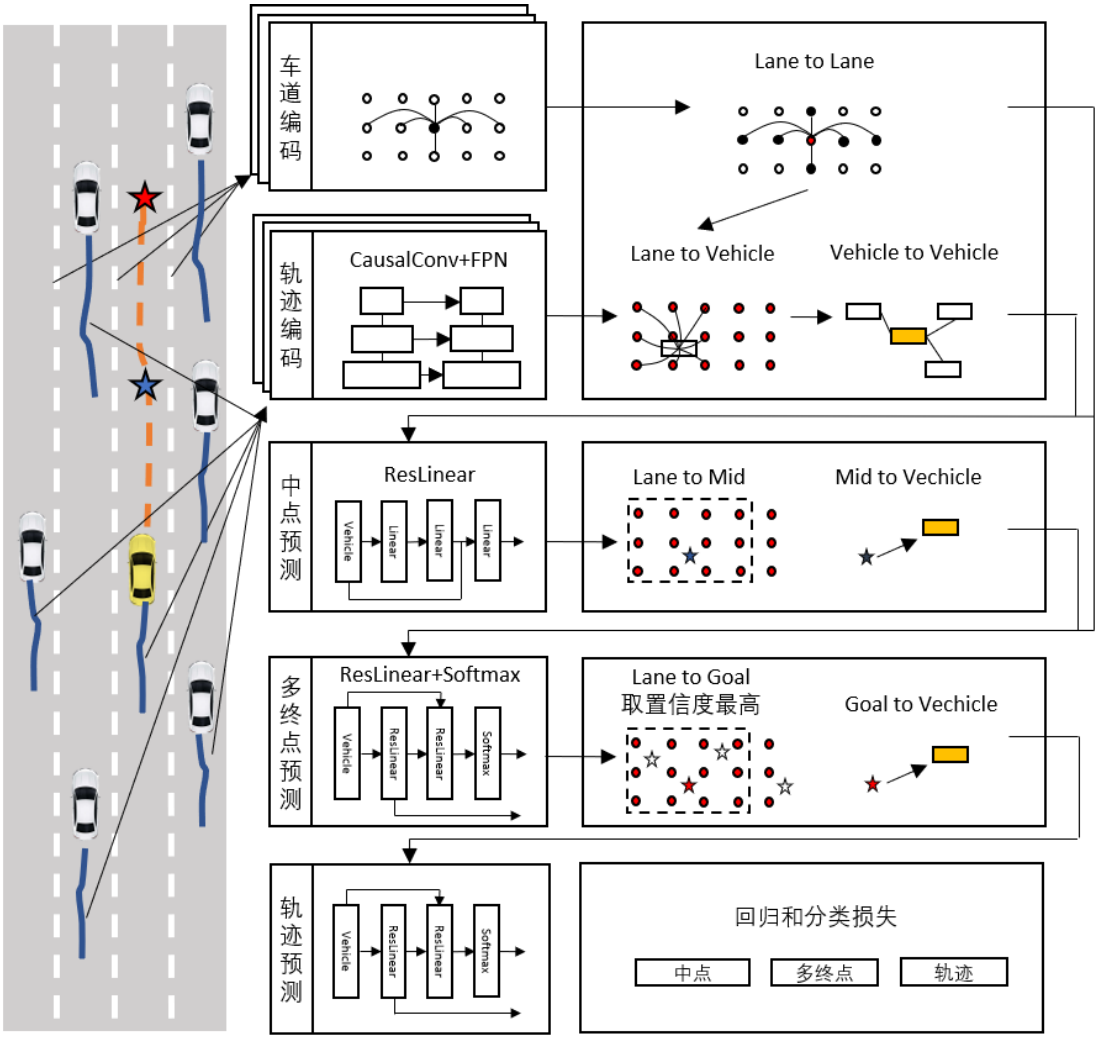


图 4.2 轨迹预测 TGCN 结构图

（1）模型输入如 4.1 节所述为车辆历史轨迹和车道信息，通过设计的两个编码器分别对轨迹信息进行特征提取，和使用 LaneGCN[56]的方法对车道线特征进行聚合。

（2）将车辆特征与车道特征（作为约束特征）进行融合，具体分为三部分，如图中 Lane to Lane 部分所示，首先是车道中心线的特征点再次聚合，相当于将较远的车道中心线特征传递给近处；Lane to Vehicle 是将前者车道线的特征作为车辆运动的环境约束与车辆特征进行融合；Vehicle to Vehicle 部分则表示车车交互，具体由空间注意力机制实现。

（3）借鉴 TNT[57]等算法，先终点预测再回归轨迹优于直接回归轨迹，所以本章也采用同样的思想，但是首先直接对中点进行预测，如图 4.2 中的蓝色星标，因为在

长时预测中，其关键转折点多以中点为主。该模块直接输出预测的当前场景中每辆车的中点位置坐标，除此之外，孤立的中点并没有包含更多车道线约束信息，所以进一步的筛选中点一定范围内的路点特征，作为中点环境信息，将该特征信息赋予交通参与者，自此每个车辆特征包含三部分信息：自身历史轨迹特征，起点车道线约束，以及中点车道线约束。

（4）在第 3 步中只预测了一个中点，但是比如在路口处中点对应的通常是所有预测的平均值，所以为了预测目标车辆左转、右转和直行的所有可能性，多终点预测也就对应解决多模态问题。该模块输出 k（=6）个终点及其置信度，并将置信度最高的终点环境特征再次赋予车辆。

（5）最后将融合的特征解码，回归出完整的预测轨迹，共包含 k（=6）条完整轨迹和对应的置信度。训练损失是针对（3），（4），（5）中预测的位置与真实位置间的误差，包括中点误差、终点误差与轨迹误差作为回归损失，而置信度的训练则采用分类损失的方法。

4.3.1 历史轨迹编码

要提取车辆本身的特征，就需要从历史轨迹中学习。本文采用因果卷积（Causal Convolution）对历史轨迹的时序特征进行提取，如图 4.3 所示，不同于常规卷积提取的是该时刻点前后范围的所有信息，因果卷积提取的是该时间点之前的信息，这也更符合实际，因为任意时刻是不可能得到具体的未来数据，而只能对历史数据进行处理。

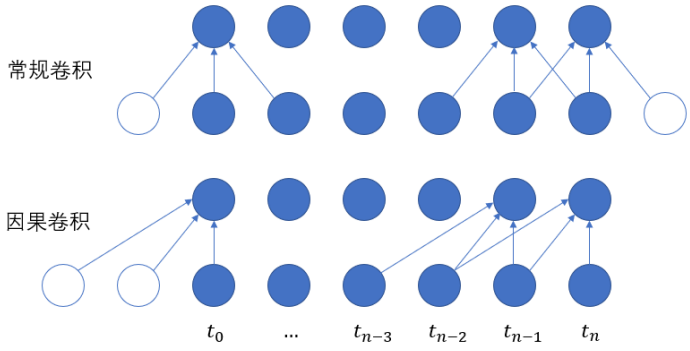


图 4.3 常规卷积与因果卷积示意图

与 2.2.1 类似使用残差结构对历史轨迹特征分三层提取，每一层为 ResBlock 残差模块，具体如图 4.4 中左框内所示，其中 Causal Conv 为上述的因果卷积其参数设置中 3 为卷积核尺寸，C 为输出通道（Channel），S 为卷积步长（Stride），GN 为组归一

化（Group Normalize），ReLU 为激活函数。其中第一层卷积步长设置为 1，主要用来改变特征通道数，输入输出尺寸一致时，就不再需要对输入额外增加一层卷积来改变拼接的尺寸，只保留分支结构但无蓝色模块。后两层用来压缩尺寸扩展维度，不断提高感受野，从而聚合更长的历史轨迹，所以输出通道数分别设置为 64，128，卷积步长设置为 2，对应左图中灰色模块，此时蓝色模块也会保留。

为了融合多尺度特征，采用 FPN 特征金字塔结构，如图 4.4 右侧所示，将三层从局部到全局的特征进行融合，右图中每层 ResBlock 模块参数定义的是输出通道数，以及卷积步长，其中卷积步长 S 具体改变的是模块中灰色和蓝色部分，其余则保持不变。Conv 使用常规卷积，这部分是为了让多尺度通道数保持一致进行的操作。Upsample 为上采样，使相邻两层尺寸保持一致从而可以进行相加操作。在最后部分再次添加 ResBlock 模块目的是学习将时序长度压缩成单位值，即每一辆车只用 128 维的特征向量来表示其历史轨迹特征。

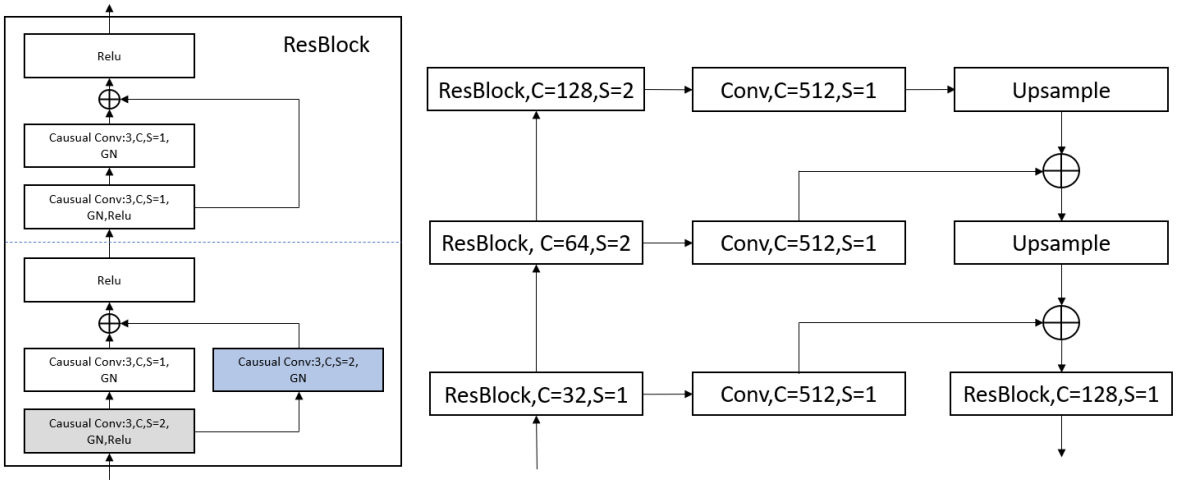


图 4.4 历史轨迹编码

4.3.2 车道环境编码

对于车道中心线环境的编码，首先需要明确中心线节点的表示方式，由位置信息（两端点的中点 v_i）和形状信息（端点相减组成的向量 $v_i^{end} - v_i^{start}$）两部分组成，所以每个路点的特征也由这两部分加和而成，如图 4.5 所示，其中 Linear 都为输出通道数为 128 维的全连接层：

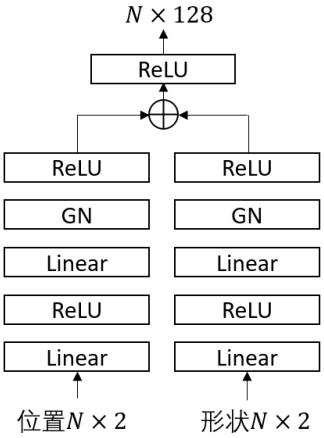


图 4.5 路点特征提取

公式如下：

$$x_i = MLP_{shape}(v_i^{end} - v_i^{start}) + MLP_{loc}(v_i) \tag{4.6}$$

在明确车道中心线每个路点的特征后，需要在更大范围内依照图结构的方式来聚合路点信息，本节采用 LaneConv[56]的车道线卷积方法，具体来说，对于车道线的图结构容易区分成三部分，如图 4.6 所示，当前位置的自身路点特征（绿色部分），横向相关的左右路点特征（橙色部分），纵向相关的前后路点特征（蓝色部分）。这里区分横纵向而不是合并在一起是考虑到车辆行驶过程中更多的是保持当前车道行驶，所以横向跨车道通常意味着一种可越过的软约束，而纵向则代表的是可保持的方向引导，所以应该用不同的权重系数表示。

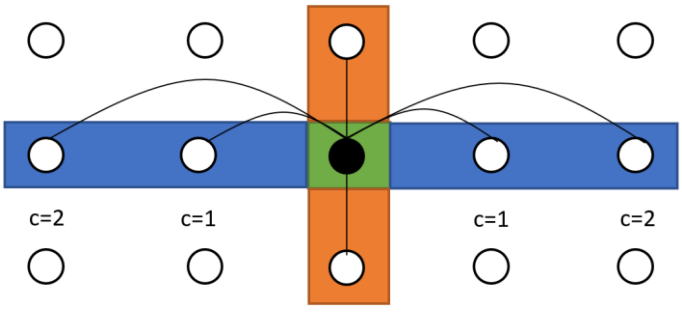


图 4.6 车道线卷积示意图

除此之外，车辆行驶的横向距离较短，但纵向距离跨度会很大，因为在短暂的时间内，车辆不会产生过大的横向位移，但是在沿着车道线的纵向方向则会前进到相对很远的位置。所以借鉴空洞卷积的概念引入空洞车道卷积（Dilated LaneConv）来实现聚合不同距离的纵向路点特征：

$$Y = XW_0 + A_{pre}^{c} XW_{pre,c} + A_{suc}^{c} XW_{suc,c} \tag{4.7}$$

其中 X 为路点特征矩阵，W 为对应的权重矩阵，A_{pre}^{c} 表示前向（predecessor）邻接矩阵的 c 次幂，意味着路点的特征信息前向传递了 c 次，从而聚合了 c 距离远的路点特征，同理 A_{suc}^{c} 表示后向（successor）邻接矩阵的 c 次幂。

在添加横向路点后，公式更新如下：

$$Y = XW_0 + \sum_{i\in\{l,r\}} A_i XW_i + \sum_{c=1}^{C} (A_{pre}^{c} XW_{pre,c} + A_{suc}^{c} XW_{suc,c}) \tag{4.8}$$

式中第一项为路点自身特征，第二项是对左右邻接路点进行聚合，第三项是对前后向节点进行聚合，C 代表的是传递范围或者空洞大小。

经过空洞车道卷积之后，通过输入输出通道数同为 128 的全连接层进行融合，再使用残差结构将输入输出进行加和，最后通过激活函数得到最终车道的编码，该模块共堆叠了 4 次，如图 4.7 所示。

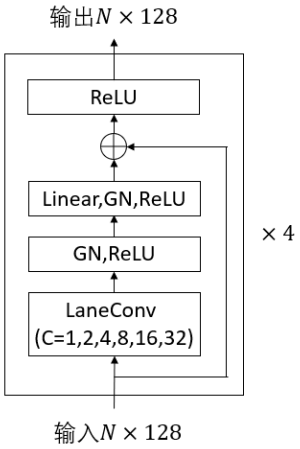


图 4.7 车道环境编码结构

4.3.3 融合与交互模型

融合与交互模型对应的是图 4.2 右上角的模块，分为三个部分：Lane to Lane，Lane to Vehicle，Vehicle to Vehicle，分别对应如下过程：首先对车道中心线的特征进行传播与更新，然后将提取的车道中心线特征作为隐式约束赋予各个车辆，最后在约束下考虑车与车之间的交互作用，最终输出的每个车辆特征融合了自车历史信息，车道中心线约束，以及交互下的响应。

（1）Lane to Lane 与 4.2.2 节所述模型相同，但此处的目的是用来传播和更新已

聚合的车道中心线特征。

（2）Lane to Vehicle 是将已更新的车道中心线特征赋予参与交互的各个车辆，该过程采用空间交叉注意力机制实现，如图 4.8 所示，其中目标向量（Target）在本环节对应的是车辆特征，环境信息（Context）对应的则是车道特征。如果对整个地图进行 Query 查询则计算量过于庞大，所以首先依据到目标车辆最后观察时刻位置，也即预测起点的位置的欧式距离，筛选出目标范围的车辆（作为 Query）和车道中心线相关路点（作为 Key），即图中 Dist Filter 模块，筛选后分别对应 Target_1 和 Context_1，范围外的则进行 mask，又因为两者维度不相同所以添加全连接层进行变换。对于筛选后的相对距离通过位置 embedding 作为位置编码。将三者拼接后的结果再进行一次 MLP 全连接层学习从而学习车道与车辆之间的关系，将车道特征赋予到车辆特征中，在衔接车辆本身的特征（看作 value），最终对应图中橙色部分，公式如下：

$$y_i = x_i W_0 + \sum_j \phi(concat(x_i, \Delta_{i,j}, x_j)W_1)W_2 \tag{4.9}$$

其中 x_i 是第 i 个目标点的特征，x_j 是第 j 个路点的特征，$\Delta_{i,j}$ 表示通过位置 embedding 之后的距离特征，具体由两层线性层与 ReLU 得到，W 是权重矩阵，通过 Linear 线性层实现，ϕ 是 MLP 全连接层，包含了四层 Linear 线性层，每层之间进行归一化和非线性激活

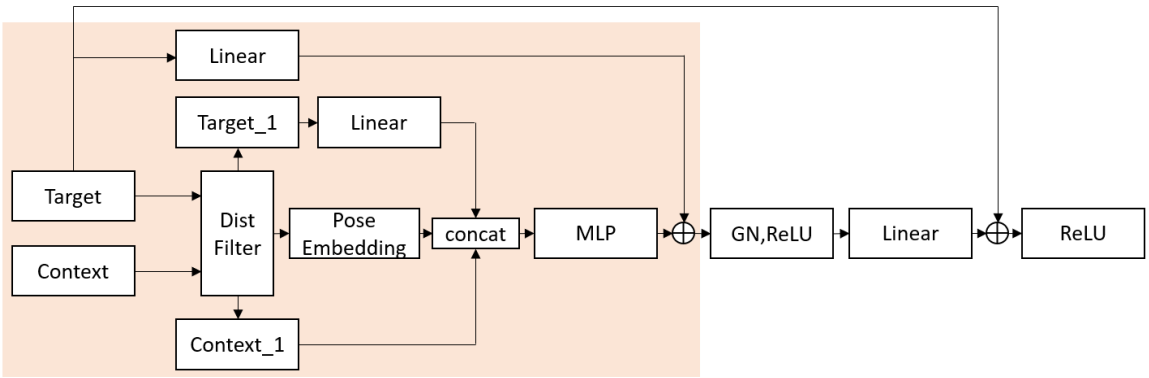


图 4.8 空间交叉注意力结构

对于最终更新的车辆特征，使用残差结构来保证输出特征至少是恒等映射，对应图中外层的结构。

（3）Vehicle to Vehicle。是将车车之间的交互特征进行相互传递，其结构与上图图一致，但此时目标向量是各车辆特征，环境信息也对应的是各车辆特征，此时该模块可以看作为空间自注意力机制，公式的输入中，x_i 则表示是第 i 个车辆的特征，

x_j是表示第j个车辆的特征。

4.3.4 目标点预测

目标点预测是通过现有的融合特征以“直接”的方式回归得到期望点位置坐标，“直接”的优点在于，该模块的输出经过最终损失函数求得的距离误差会通过网络的反向传播直接作用于该模块，从而使得目标点预测更加准确，同时让后续预测的轨迹被限制在起点与目标点之间，得到更小的平均误差，也即更高的准确率。

目标点预测在本节分成了中点预测和多终点预测两个流程。前者只预测未来轨迹的一个中点，而后者是预测6个终点及其对应置信度。不同于单终点的输出，相当于其置信度为1，从而完全信赖该预测结果，多终点（多模态）预测的现实意义是可以给后续规划以不同置信度的动态障碍空间，对于未来不确定的它车轨迹可以更安全或更谨慎的作为动态障碍物。

中点预测是将融合特征通过如图 4.9 所示的 ResLinear 残差结构下的全连接层解码，输出中点位置坐标(x, y)。

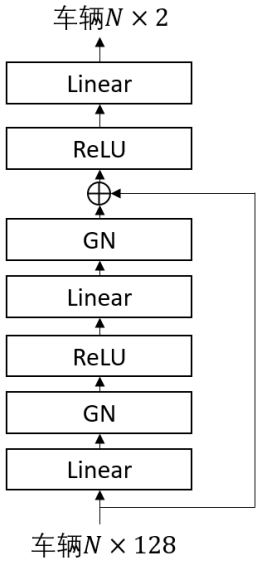


图 4.9 中点预测 ResLinear 结构

不同于 TNT 等算法，中点同样可以作为一个隐式约束的赋予每个参与者的特征中，但是该中点同样不能脱离对应位置范围内的车道中心线的实际条件，所以本节再次采用与图 4.8 相同的空间交叉注意力方法，此时 Dist Filter 模块筛选的是与中点位置相对距离在阈值范围内的车道中心点，从而将中点处的车道约束赋予给车辆特征中。

多终点预测对应的是多模态输出，预测的是 6 个终点，如图 4.10 所示。对于每个终点$(x_i, y_i), i = 1,2,...,6$，使用与中点预测相同的解码子网络，如图中蓝色部分，共进行了 6 次。而要计算每个终点的置信度则由后两部分完成，除了车辆特征外，还需要对起终点相对距离进行位置编码，如图中绿色部分，将两者拼接后使用 ResLinear 结构解码输出，如图黄色部分，最后通过 Softmax 得到对应每个终点的置信度。

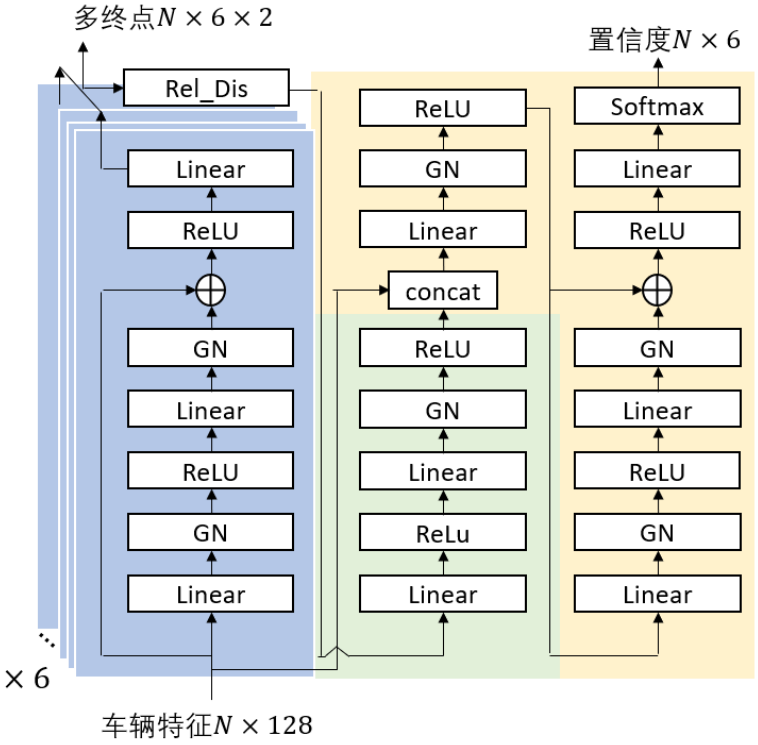


图 4.10 多终点预测结构

因为对于未来明确发生的实际只有一个真值，即实际只有一条未来轨迹，所以并不能对所有终点明确求得对应多真值的误差损失，就需要对终点的置信度进行排序，以置信度最高的终点作为当前模块所关注的终点位置，而其余五个终点则在大量数据中自动学习成其他模态所对应的终点，比如在进入岔路口前，预测的多终点可能是直行，也可能是转弯，也可能更远也可能更近，但总是期望置信度最高的，比如直行的终点为当前更关注的终点，其它的转向终点信息只作为包含了未来的可能性用以提供参考。这些多终点并不需要在模型初始就对场景进行分类处理，即从一开始去判断转向还是直行，加速还是减速等，只要数据量足够大，搭建的神经网络会自动学习潜在的类别，同时这也才能在设计损失函数时与真值对比，否则将所

有结果与真值对比学习将会使得预测结果趋同，失去多模态本身的意义。

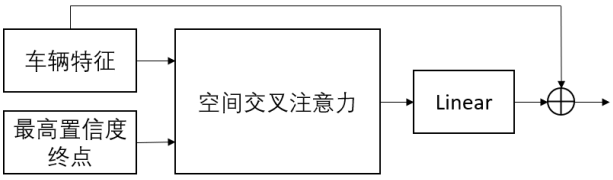


图 4.11 多终点特征融合结构

对于选取的置信度最高的终点，再次使用 Dist Filter 模块，筛选与置信度最高的终点的相对距离在阈值范围内的车道中心点特征，并将终点处的车道约束赋予车辆特征中。

至此，输出的对应特征实际包含：1、车辆自身特征；2、起点附近车道中心线下的潜在约束特征；3、车车交互的特征；4、目标点附近车道中心线下的潜在约束特征；

4.3.5 轨迹预测解码

本节则是进行特征解码，输出预测的完整轨迹及其对应置信度。

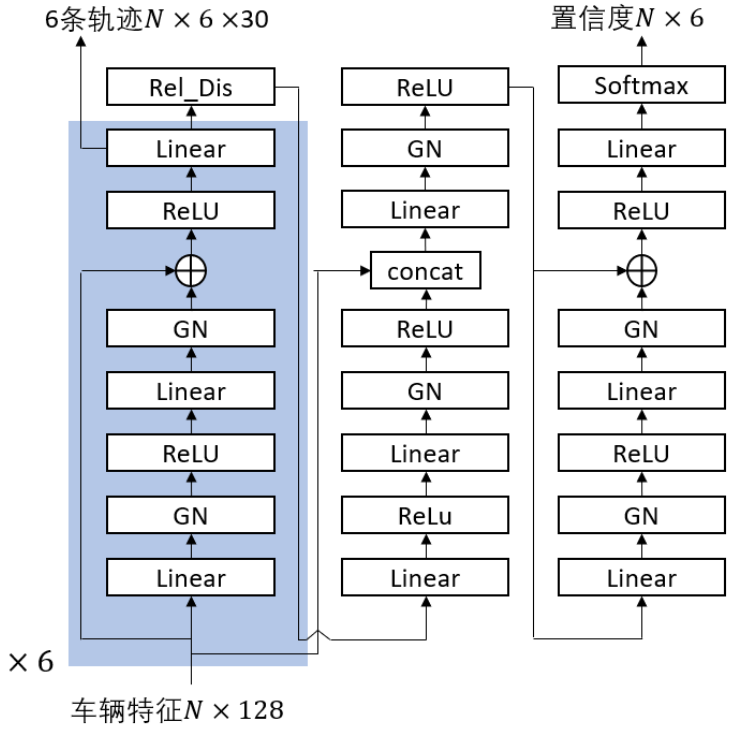


图 4.12 轨迹预测解码结构

解码结构与多终点预测类似，主要区别在于特征解码输出的是完整的轨迹共

$30\times(x,y)$个轨迹点，而 6 条轨迹共需要解码 6 次，最终得到$(6,30,2)$维度的预测值，如图 4.12 蓝色部分。每条预测轨迹所对应的置信度采用与求解终点置信度相同的方式，如图中白色部分。

因为真值轨迹存在波动，所以最终预测的轨迹点也并不平滑，进一步的采用 Savitzky-Golay 滤波算法对预测轨迹进行平滑处理，该算法是基于局部最小二乘多项式逼近的数据平滑方法，能够在滤波的同时保证信号原本形状不变，主要思想是对一定长度窗口内的数据点进行 n 阶多项式拟合，所以是一种移动窗口式加权平均算法，但加权系数是在窗口内对给定高阶多项式的最小二乘拟合得到。

假设滤波窗口包含长度为 2m+1 的连续的离散数据$x[i],i=-m,...,0,...,m$，构造$n(\leq 2m+1)$阶多项式进行拟合，有：

$$f(i)=\sum_{k=0}^{n}b_{nk}i^{k} \tag{4.10}$$

拟合数据与原数据的残差平方和为：

$$E=\sum_{i=-m}^{m}(f(i)-x[i])^{2}=\sum_{i=-m}^{m}\left(\sum_{k=0}^{n}b_{nk}i^{k}-x[i]\right)^{2} \tag{4.11}$$

为使残差平方和最小，对 E 中的多项式的各系数b_{nr}求偏导使之等于 0：

$$\frac{\partial E}{\partial b_{nr}}=2\sum_{i=-m}^{m}\left(\sum_{k=0}^{n}b_{nk}i^{k}-x[i]\right)i^{r}=0,r=0,1,...,n \tag{4.12}$$

可解得：

$$\sum_{k=0}^{n}b_{nk}\sum_{i=-m}^{m}i^{k+r}=\sum_{i=-m}^{m}x[i]i^{r} \tag{4.13}$$

令：

$$\left.\begin{aligned}F_{r}&=\sum_{i=-m}^{m}x[i]i^{r}\\S_{k+r}&=\sum_{i=-m}^{m}i^{k+r}\end{aligned}\right\}\Rightarrow F_{r}=\sum_{k=0}^{n}b_{nk}S_{k+r} \tag{4.14}$$

给定拟合的单边点数 m，多项式阶数 n 和待拟合数据 x，可以求得F_{r}，再将S_{k+r}代入(4.14)可以求得多项式的各系数$b_{nr},r=0,1,...,n$，从而可以确定$f(i)$拟合的多项式。

4.3.6 损失函数

损示函数包含三项内容：中点预测损失，多终点预测损失，轨迹预测损失。

（1）中点预测损示 L_{mid}，因为只预测每辆车的一个中点位置，所以只需要求相对欧式距离作为损示即可，最终求所有预测平均值：

$$L_{\text{mid}} = \frac{1}{N}\sum_{n=1}^{N} smoothL1(p_{\text{mid,n}} - p_{gtm,n}) \tag{4.15}$$

式中 $p_{\text{mid,n}}$ 表示第 n 辆车的中点预测位置，$p_{gtm,n}$ 表示第 n 辆车的真实中点位置，smooth L1 loss 损示表示如下：

$$smoothL1(x) = \begin{cases} 0.5x^2 & if \left\| x \right\| < 1 \\ \left\| x \right\| - 0.5 & otherwise \end{cases} \tag{4.16}$$

（2）多终点预测输出 6 个终点及其置信度，所以包含分类和回归两部分损失，分类损失采用最大边缘损失，因为期望选取的置信度是最高置信度，所以初始定义 $\hat{k}$ 标记该值，而其他终点对应的置信度不会超过所选取的最优终点，即计算其它 k 模态所属值与 $\hat{k}$ 的差值，来最大边缘化期望置信度：

$$L_{multi,cls} = \frac{1}{N(K-1)}\sum_{n=1}^{N}\sum_{k\neq\hat{k}} \max(0, c_{multi,k} + \varepsilon - c_{multi,\hat{k}}) \tag{4.17}$$

式中 $c_{multi,k}$ 表示多终点预测的 k 模态对应的置信度值，ε 表示定义的裕度，本文设计为 0.2，max 表示如果其他置信度比所选的最高置信度小于改阈值，则记为 0，若大则会产生损失值，需要反向传播优化，从而确保后续置信度最高的终点为当前所选的“最优”终点。

回归损失则与(4.15)相同：

$$L_{multi,reg} = \frac{1}{N}\sum_{n=1}^{N} smoothL1(p_{multi,n}^{\hat{k}} - p_{gt,n}) \tag{4.18}$$

式中 $\hat{k}$ 表示选择置信度最高的终点，$p_{gt,n}$ 表示第 n 辆车的真实终点位置。

$$L_{multi} = L_{multi,cls} + \alpha_1 L_{multi,reg} \tag{4.19}$$

式中 $\alpha_1 = 0.2$

（3）轨迹预测损失，针对的是多模态预测的整条轨迹，同样包括分类损失和回归损示，与前者类似有：

$$L_{traj,cls} = \frac{1}{N(K-1)} \sum_{n=1}^{N} \sum_{k \neq \hat{k}} \max(0, c_{traj,k} + \varepsilon - c_{traj,\hat{k}}) \tag{4.20}$$

$$L_{traj,reg} = \frac{1}{NT} \sum_{n=1}^{N} \sum_{t=1}^{T} smoothL1(p_{traj,n,t}^{\hat{k}} - p_{gt,n,t}) \tag{4.21}$$

$$L_{traj} = L_{traj,cls} + \alpha_2 L_{traj,reg} \tag{4.22}$$

式中同样 $\alpha_2 = 0.2$。

最终总损失表示为：

$$L = \beta_1 L_{mid} + \beta_2 L_{multi} + \beta_3 L_{traj} \tag{4.23}$$

其中 L_{mid} 最作为最初的中点特征融合到编码特征中，对后续影响较大，另外中点预测包含更多不确定的因素，所以设置 $\beta_1 = 2$，而后续的损示设置为 $\beta_2 = \beta_3 = 1$

4.4 实验结果与分析

4.4.1 实验数据集

Argoverse 数据集[54]是由 Argo AI、卡内基梅隆大学和佐治亚理工学院发布自动驾驶数据集，其采集平台包括 2 台 32 线激光雷达传感器，7 个高分辨率环形摄像头，和 2 个前置立体摄像机。

图 4.13 Agroverse 数据采集平台

其中 Motion Forcecasting 轨迹预测数据集，包含了超过 1000 驾驶小时的数据中提取的 327790 个有价值的场景，并拆分成 208272 个训练序列，40127 个验证序列和 79391 个测试序列，其中每个场景中包含了自动驾驶车辆 5 秒钟的行驶轨迹，具体用 2 秒预测未来 3 秒，同时还跟踪了其它车辆的轨迹。而高精地图数据集包含了 290KM 的带有几何形状和语义信息的高精地图数据，包括车道中心线及其属性的矢量图、地面高度的栅格化地图和可行驶区域和感兴趣区域的栅格化地图。

表 4.1 Argoverse 预测数据释义

字段	字段长度	单位	含义
TIMESTAMP	1	秒	时间戳，采集频率为 10Hz
TRACK_ID	1		每辆车独有的 ID
OBJECT_TYPE	1		交通参与者类别
X	1	米	X 坐标
Y	1	米	Y 坐标
CITY_NAME	1		城市名称，用来对应不同地图

4.4.2 评价指标

主要评价指标为终点位移误差 FDE（Final Displacement Error），表示的是终点预测位置和终点真值位置的平均欧式距离，平均位移误差 ADE（Average displacement Error），表示的是预测轨迹中每个位置和真值位置之间的平均欧式距离，未命中率 MR（Miss Rate）表示预测的终点超出真实终点阈值（2m）范围外所占的比例。根据预测轨迹输出条数，又分为单模态和多模态预测，前者只预测一条轨迹（K=1），后者预测多条轨迹（K=6）。此外 brier_minFDE 表示的是综合考虑置信度和终点误差的评价指标。

（1）单模态

$$FDE = \frac{1}{N}\sum_{n=1}^{N}(p_{end,n} - gt_{end,n})^2 \tag{4.24}$$

$$ADE = \frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}(p_{t,n} - gt_{t,n})^2 \tag{4.25}$$

$$MR = \frac{card((p_{end,n} - gt_{end,n})^2 > 2)}{N}, n = 1,...,N \tag{4.26}$$

式中 $p_{end,n}$ 表示第 n 辆车预测的终点位置，$p_{t,n}$ 表示第 n 辆车在第 t 时刻的位置，gt 表示真值，$card(\bullet)$ 表示集合中满足条件的个数

（2）多模态

与单模态输出的区别在于，只计算所有预测轨迹中最接近真值的作为预测结果，所以对应指标数值也会小于单模态指标：

$$mFDE = \min_{k}(\frac{1}{N}\sum_{n=1}^{N}(p_{end,n,k} - gt_{end,n})^2), k \in K \tag{4.27}$$

$$mADE = \min_{k}(\frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}(p_{t,n,k} - gt_{t,n})^2), k \in K \tag{4.28}$$

$$mMR = \min_{k}(\frac{card((p_{end,n,k} - gt_{end,n})^2 > 2)}{N}), n = 1,...,N; k \in K \tag{4.29}$$

式中 K 表示预测的轨迹数，本文中为 6 条，$p_{end,n,k}$ 表示第 n 辆车预测的第 k 条轨迹的终点位置，$p_{t,n,k}$ 表示第 n 辆车预测的第 k 条轨迹在 t 时刻的位置，gt 表示真值，min 表示只保留 K 个结果中最小值。

（3）brier_minFDE

对于多条预测的轨迹，需要考虑其概率校准，而该指标就是综合了距离误差与概率的度量，该指标值越小表明预测结果越好：

$$b\_mFDE = \min_{k}[(\frac{1}{N}\sum_{n=1}^{N}(p_{end,n,k} - gt_{end,n})^2)(1 - p_k)^2], k \in K \tag{4.30}$$

式中 p_k 表示的是第 k 条轨迹对应的置信度或概率值。

4.4.3 实验结果与对比

（1）实验结果可视化

可视化结果如图 4.14-图 4.19 所示，分成直行、路口和异形 路口三种不同的工况，其中目标车辆的历史轨迹用蓝色线标出，起终点分别由红色的圆和三角形标出，对应的多轨迹输出由绿色实线表示，真值由红色实线表示，车道中心线由虚线表示。

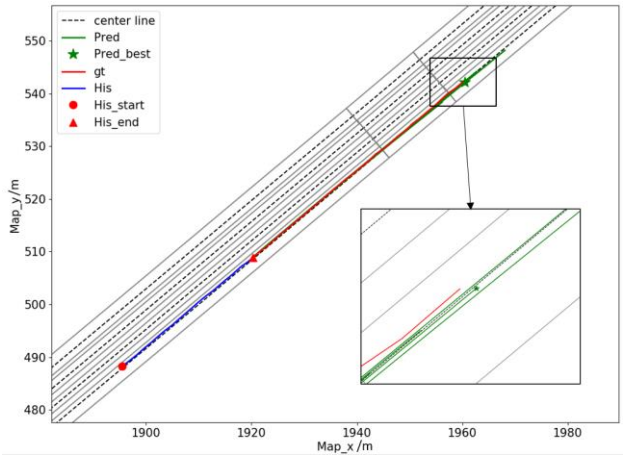


图 4.14 直行工况轨迹预测

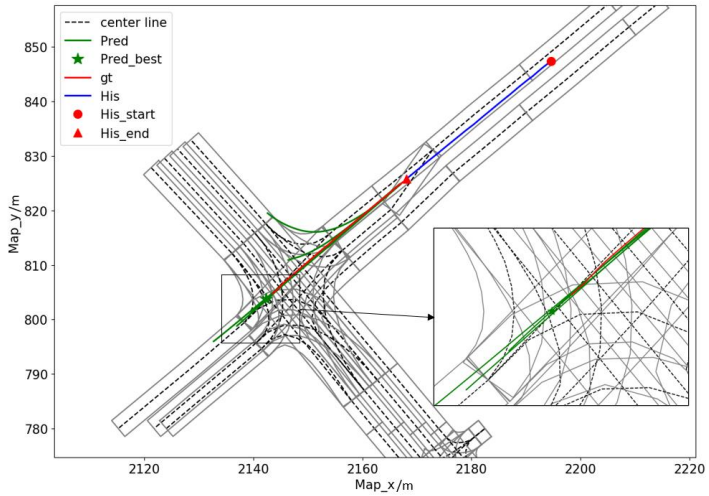


图 4.15 路口工况轨迹预测（一）

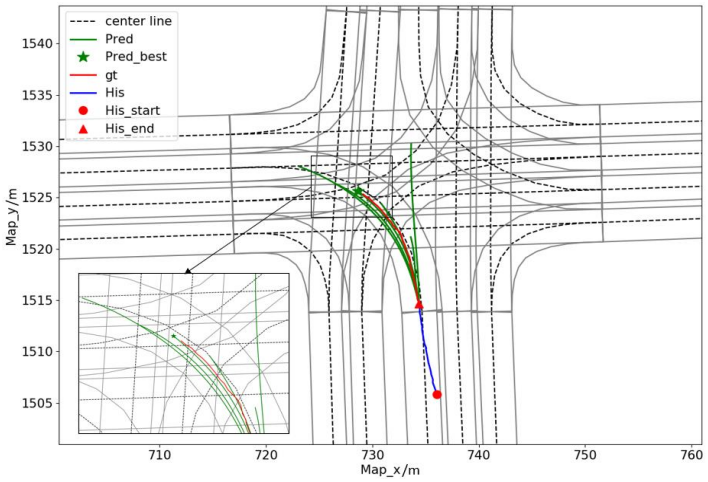


图 4.16 路口工况轨迹预测（二）

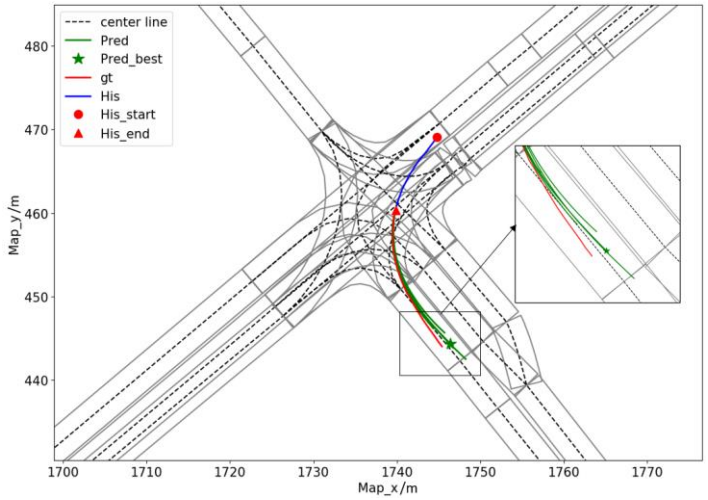


图 4.17 路口工况轨迹预测（三）

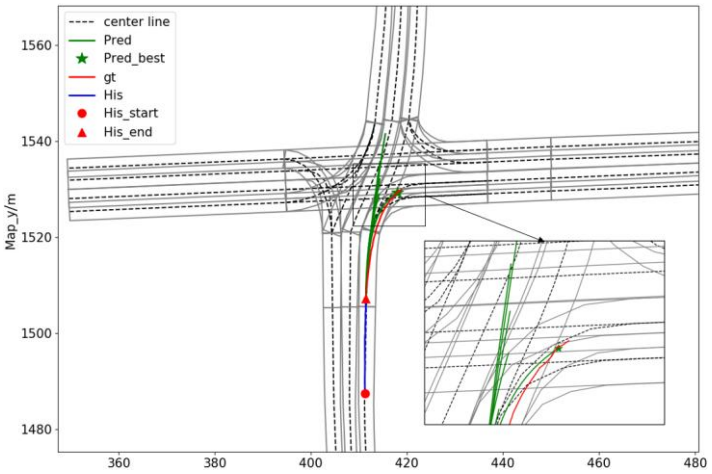


图 4.18 异形路口轨迹预测（一）

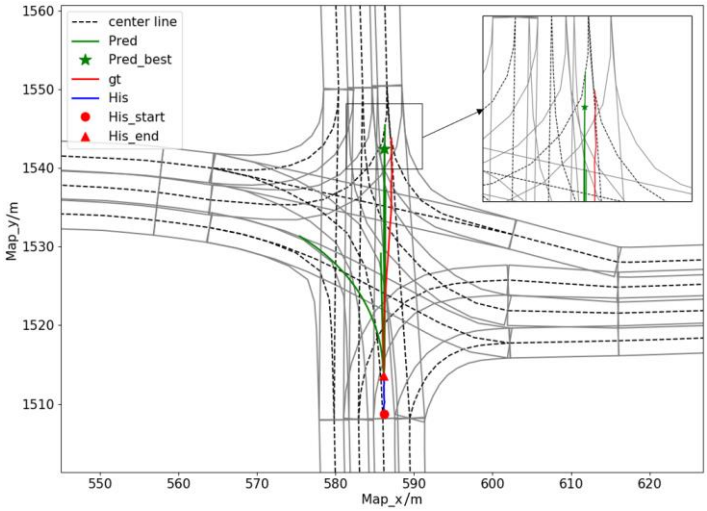


图 4.19 异形路口轨迹预测（二）

在直行工况中，如图 4.14 所示，预测的六条轨迹全部都符合直行，其中最优预测也在真值附近。在常规的路口工况中，如图 4.15，此时车辆虽然还未进入路口，同时也没有进入左转道路中时，预测也只分成了直行和右转两种模态，其中主要的预测都以直行为准，而真值也确实是直行通过路口；而在图 4.16 中，目标车辆刚进入路口，因为车道线属性限制，此时车速较低，预测也以不同程度的左转为主，但值得注意的是，其中一条预测的轨迹也没有排除直行的可能，这是因为在实际情况中（在大量数据学习下），并非所有车辆都会遵守交通规则，总会有例外，体现出预测的多模态可能性，就最终预测结果而言，最可能的还是左转，也符合真值表现；相比之下，在图 4.17 中，车辆已行驶在路口中央，依据其历史轨迹和所处环境，所

有预测都判断为左转这一种可能，而在实际该车也是左转。从上可以看出，对于目标车辆所处路口的位置不同，预测也会由多种可能性逐渐变为最可能的那一种，同时即使在不同位置，最主要的预测也基本能预测准确。在异形路口中，如图 4.18 和图 4.19 所示，得益于所建立的道路图结构，提出的模型也能实现完成复杂工况下的轨迹预测任务，同时保持较好的预测结果。

（2）实验结果对比

依据各项评价指标，对比各种算法在 Argoverse 测试集中的测试结果如下表所示：

表 4.2 Argoverse 测试集结果对比

方法	b-minFDE (K=6)	mFDE (K=6)	mADE (K=6)	MR (K=6)	mFDE (K=1)	mADE (K=1)	MR (K=1)
LaneRCNN[78]	2.147	1.453	0.904	0.123	3.692	1.685	0.569
TNT[57]	2.140	1.446	0.910	0.166	4.959	2.174	0.710
LaneGCN[56]	2.059	1.364	0.868	0.163	3.779	1.706	0.591
mmTransformer[79]	2.033	1.338	**0.844**	0.154	4.003	1.774	0.618
GOHOME[80]	1.983	1.450	0.943	0.105	3.647	1.689	0.572
HOME[59]	-	1.450	0.940	**0.102**	3.730	1.730	0.584
DenseTNT[58]	1.976	**1.282**	0.882	0.126	3.632	1.679	0.584
TGCN(终点) (proposed)	2.026	1.344	0.874	0.149	3.593	1.654	0.574
TGCN(中点) (proposed)	**1.967**	1.286	0.861	0.136	**3.576**	**1.640**	**0.567**

本文在所提出的 TGCN 也测试了两种模型，TGCN（终点）是指将第一个目标点预测更改为对轨迹终点预测的结果，TGCN（中点）则是本章最终构建的模型。实验表明虽然过早的融入终点特征在一定程度上减少了终点距离误差，但是同样会造成多模态的趋同性，使得平均误差较高，b_minFDE 指标也较高。而中点预测的核心思想则是将长时预测拆分成两个较短的预测，前一阶段保留了车辆的原始运动特性，而将多模态的预测放入后半段中，所以结果也更好。此外，因为 TGCN 融合了 LaneGCN 的车道线编码和 TNT 的终点预测思想，可以看出对比这两种算法，在所有指标上都得到了很高的提升。在对比其他不同的主流算法中，所提出的方法在单轨迹预测效果更好，这是因为所构建的模型在中点预测和多终点预测中都加强了目标点附近的特征提取，而在多轨迹预测中也相差不大，在综合考虑距离误差和概率校

正的指标 b_minFDE 中达到了更好的水平。

4.5 本章小结

本章提出的轨迹预测模型 TGCN（Target considered Graph Convolution Network）最主要的特点是将目标点的特征融入到轨迹预测里。首先，分别从车辆的历史时序状态提取车辆运动特征，和所处车道的环境信息提取车道中心线特征。然后，将环境特征约束赋予每个交通者，另外考虑到车与车之间的交互，所以采用空间注意力机制进一步地将交互信息进行融合学习。接着，为了减少预测轨迹与实际真值的误差，直接对中点进行预测，那么轨迹预测问题就可以看作已知起点和经过中点校正后如何回归未来轨迹的问题。但是在实际环境中，尤其是在路口处，对车辆直行还是转向需要进行合理的判断，所以在中点的基础上进一步的预测多个终点，并取其中置信度最高的终点处的环境特征也融入到车辆特征内，其余终点则是在大量数据训练下，去自动学习分类成不同的多模态终点。最后，将车辆信息，约束信息，车车交互信息，起点和目标点环境信息融合特征进行解码，输出预测的轨迹。在实验结果可视化与对比中，可以看出所提出的 TGCN 模型有更好的综合表现。

第 5 章 检测跟踪实车实验与多车轨迹预测

本章针对第二章单目 3D 车辆检测和第三章的多目标车辆跟踪所搭建和训练的模型在真实城市路况进行实车测试并进行定性分析，并对第四章轨迹预测进一步拓展为多车辆轨迹预测的仿真测试并进行可视化分析。

5.1 实验设备及场地

本实验是在如图 5.1（a）所示的本田 CR-V 车辆平台上，搭载（b）所示的相机设备，并只使用左视相机以1240×370分辨率 15Hz 帧率进行视频数据采集。实验地点设计在学校周边如图 5.2 所示的绿色路线，图中标识处为主要采集路段 A-E，包含直行，路口工况，且路口处因为有复杂的人流遮挡情况，所以也适合用来进行多目标跟踪的遮挡验证。

(a)本田 CR-V　　(b)视频采集相机（只使用左目）

图 5.1 实车实验平台

图 5.2 数据采集路段路线图

5.2 多目标检测与跟踪实车实验

因为多目标车辆跟踪是以单目 3D 车辆检测为基础，所以本节实验是同时对两个算法进行联合验证的过程。本节没有进行额外的定量分析是因为对 3D 目标检测的真值数据采集除了极高的设备要求，同时也需要大量的人力时间成本来标注，所以借助已有的数据集训练模型后，从实车的测试效果进行定性分析。

图 5.3 路段 A（左）和路段 C（右）

从 3D 车辆检测角度来看，在较为复杂的场景中表现出较好的识别效果，对于较小的图像截断，以及被部分遮挡时能够完成 3D 检测任务。从跟踪角度来看，此时既有自车运动又有它车运动，在路段 A 处中白色矩形框部分，放大后如图 5.4 所示，在第 290 帧识别到紫色框标识的 ID 为 87 的白车在后续视频里被 ID 为 70 的黑车所遮挡，等经过岔路口合流到主路中后，即第 370 帧重新出现时仍能够重新跟踪到并正确标识 ID。在路段 C 处中，对比第 80 帧和第 100 帧，在由远及近的过程中，左侧的同一辆车因为过多的遮挡而对外观识别不够准确而产生了 ID 变换，但对于视频中央 ID 为 13 的白车，如图 5.5 所示，被后方黑色车持续遮挡后仍能够跟踪到。

Frame 290　　Frame 320　　Frame 370

图 5.4 路段 A 遮挡后重识别细节

Frame 90　　Frame 100　　Frame 120

图 5.5 路段 C 遮挡后重识别细节

图 5.6 路段 B（左）和路段 D（右）

路段 B 处于路口处，此时在旁车道中等待交通灯的静止车辆较为密集，检测视

角的变化也主要来自于自车运动，所以在场景相对简单，但是车辆较为密集的环境中对于可观测的车辆能够保持较好的识别效果。对于正前方处于转向中的灰色车辆也能够实现较好的跟踪。在图 5.6 的路段 D 中，处于图像中央的绿色框，ID 为 77 的车辆也在进行转向，此时自车也在不断靠近，在更为复合的场景下能够较为稳定的跟踪。

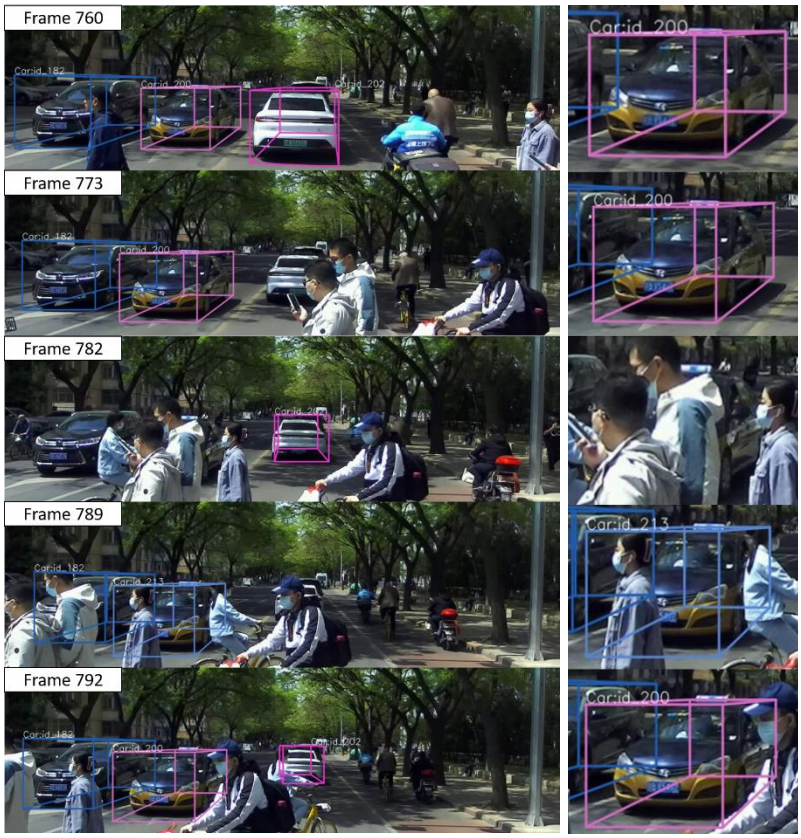


图 5.7 路段 E（左），ID 切换细节（右）

在图 5.7 所示的路段 E 中，路过的行人对目标车辆形成不同程度的遮挡，在第 760 帧中，图中三辆车都可以识别到，其中右侧白车正在驶离人群。在第 773 帧时，该车被遮挡，第 782 帧中重新出现，鉴于良好的外观特征以及在相机中较稳定的位置所以实现了重识别。而在第 789 帧中，左侧两辆车被遮挡后重新识别处位置信息，但是在图右的 ID 切换中，被粉色框所标识的 ID 为 200 的出租车，因人群的干扰了外观特征，从而被错误的认为是新的跟踪目标 ID 变为 213，而在第 792 帧，行人离开，遮挡程度降低，此时得到的外观与之前的特征库重新匹配后，变回原有 200 的 ID。可以看出人群的长时离散遮挡会影响外观特征的改变，对于重识别是个极大的挑战。

5.3 多车辆轨迹预测实验

在第四章中本文搭建了车辆轨迹预测模型，并对目标车辆实现了轨迹预测，并做了结果可视化展示，但在实际行驶过程中往往关注的不只是一辆车，而是对当前环境下所有交通参与者的未来走向都能够进行较为准确的预估，所以在本文所设计的预测模型训练中，也就没有特别针对数据集所明确的目标车辆，而是将所有的道路使用者都进行了统一的学习，因此本节是在前者的训练精度的基础上对多车辆的轨迹预测进行可视化分析。可视化结果如下图 5.8-图 5.13 所示，分成直行、路口和异形路口三种不同的工况，其中目标车辆的历史轨迹用蓝色线标出，对应的多轨迹输出由绿色实线表示，真值由红色实线表示，对于其它交通参与者只保留了置信度最高的预测轨迹，由绿色虚线表示，对应的真值则由红色虚线表示，图中所有的红色三角形都表示预测的起点。

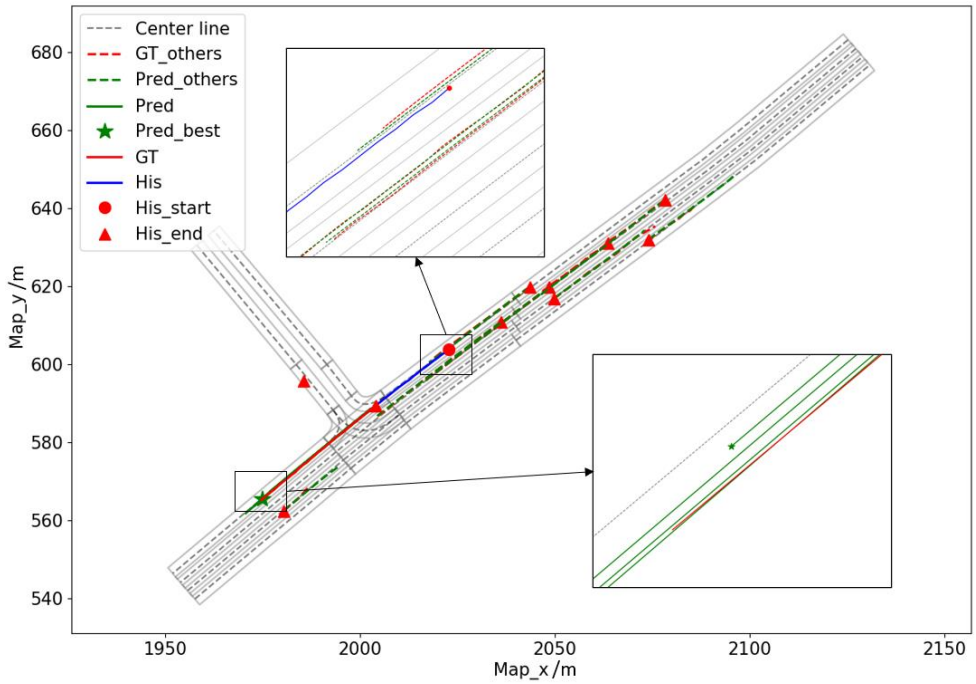


图 5.8 直行工况轨迹预测（一）

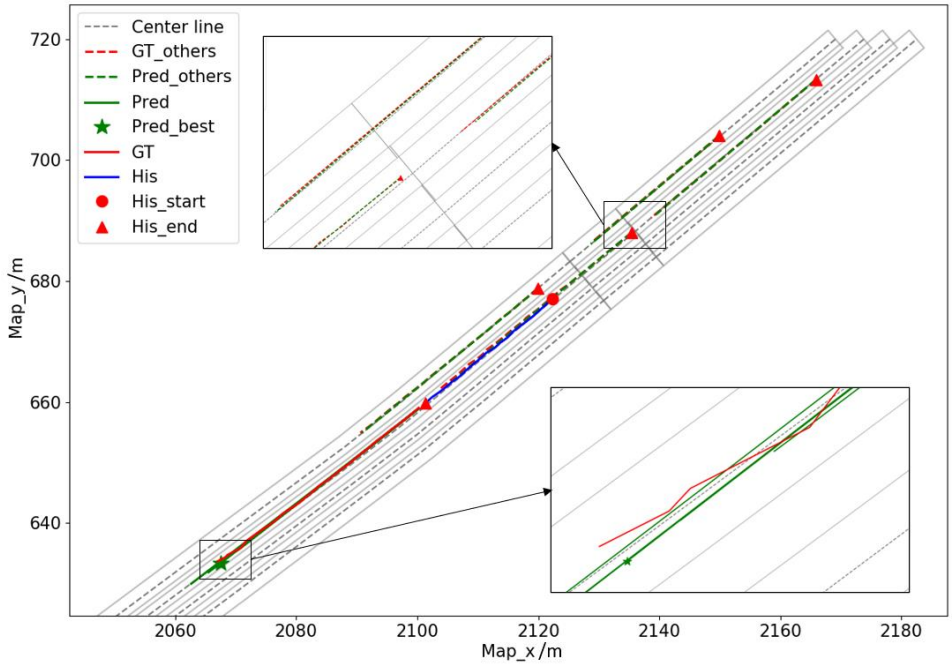


图 5.9 直行工况轨迹预测（二）

如图 5.8 和图 5.9 所示为直行工况下的多车辆轨迹预测，此时预测的轨迹也为沿着车道中心线的轨迹，与真实轨迹重合度较高，预测的终点位置和真值也比较接近。

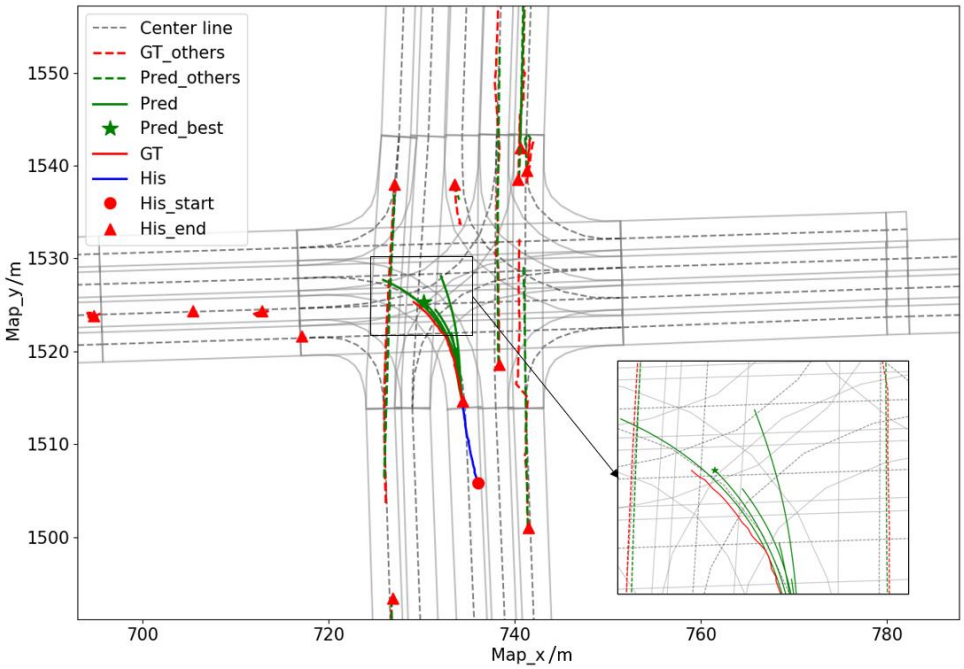


图 5.10 路口工况轨迹预测（一）

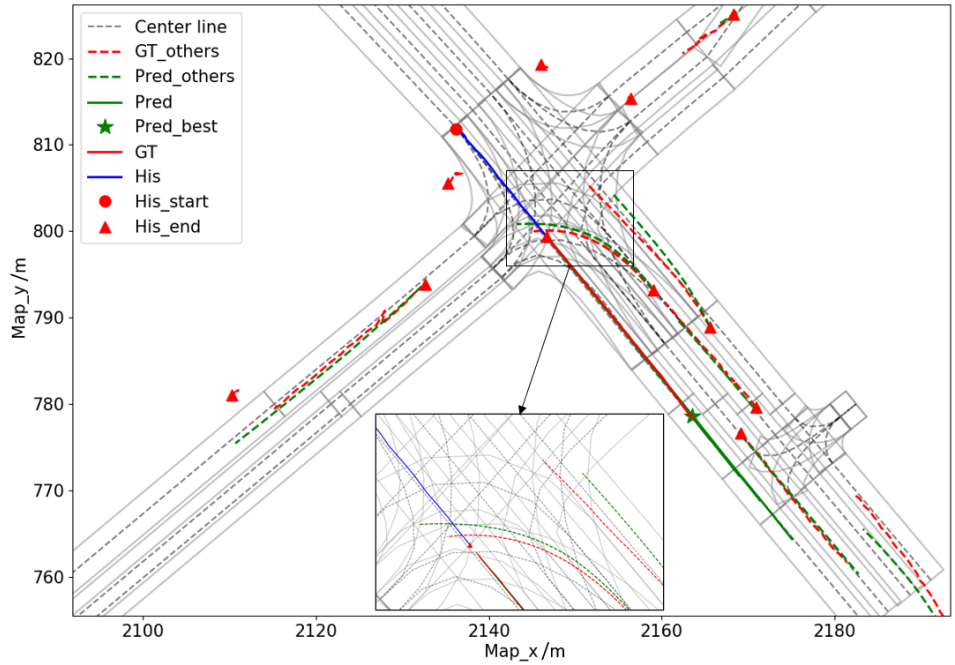


图 5.11 路口工况轨迹预测（二）

在图 5.10 中，左方车辆处于停车等待中，对应预测的轨迹也仍处于原地，被真值所覆盖，而由绿色实线标出的目标车辆可以看出，预测呈现左转和直行两种可能，实际选择为左转与真值相符，同样的，其他车辆的预测中置信度最高的轨迹都为穿越路口的直行，也与真值相符；对比在图 5.11 中，可以明显看出对左转和直行车辆轨迹的正确预测。

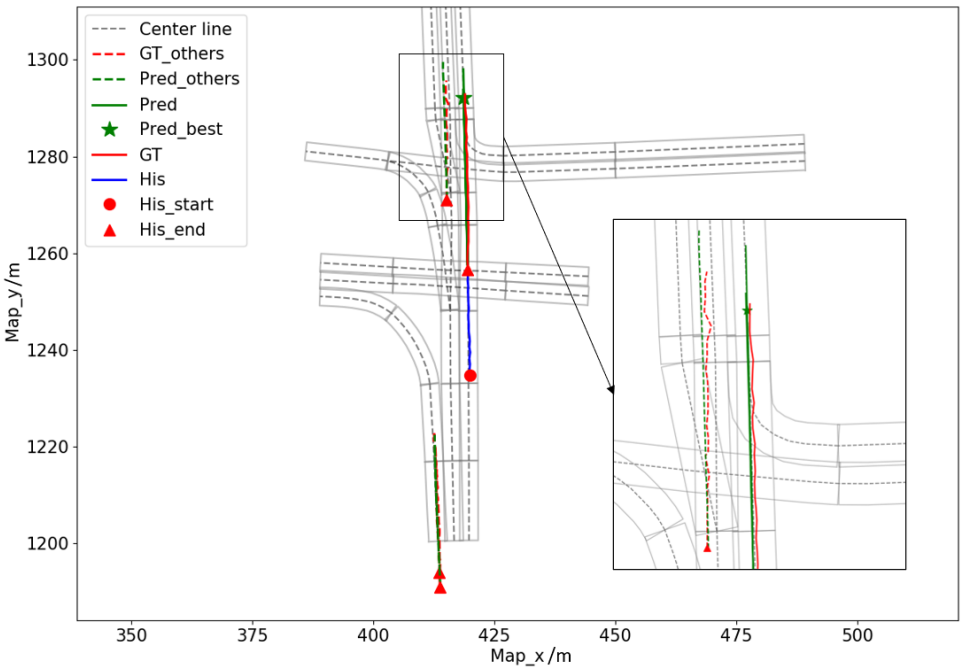


图 5.12 异形路口轨迹预测（一）

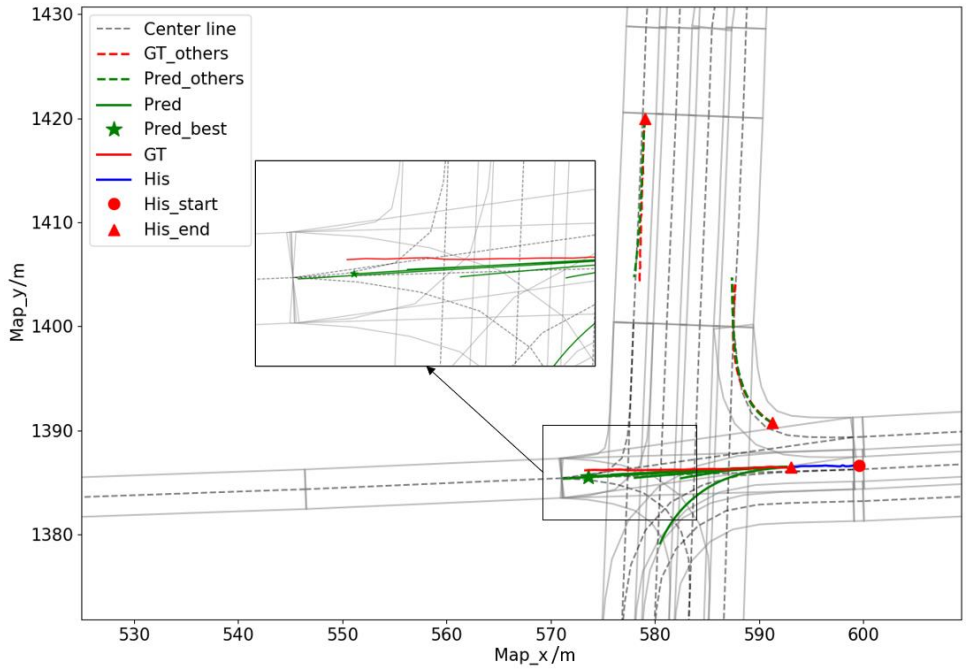


图 5.13 异形路口轨迹预测（二）

在对异形路口的轨迹预测中，图 5.12 由于车道线属性的限制，对目标车辆所预测的六条轨迹为不同距离的直行，其左车的轨迹同样为直行预测，而对于下方的车辆则是沿着车道线的方向的行驶，再次说明构建道路的关系拓扑图的重要性。在图 5.13 中对于即将进入路口，又不处于右转车道的目标车辆预测得到直行和左转两种可能，对比处于右转车道的它车也正确预测了其未来的轨迹。

5.4 本章小结

本章是对第二和第三章所提出的单目 3D 车辆检测算法和多目标跟踪模型在实际路况中进行实车测试结果定性分析，从图中可以看出在驾驶视角下能够较为精准的完成 3D 目标检测并实现稳定的多目标跟踪。而对于车辆预测模型的分析则不局限目标车辆，而是同时预测场景中所有的交通参与者，从直行、路口和异形路口的结果图可以看出，预测的轨迹与真值较为相符，对于是否转向也能够准确判断。

结论与展望

1. 总结

在自动驾驶系统中，感知的关键是识别行驶过程中周围的车辆，从而明确不可碰撞的空间区域，除此之外对这些车辆未来的行驶轨迹进行准确估计，也能更好的帮助自车进行后续的决策规划和控制。本文针对在随车视角下单目 3D 目标检测与多目标跟踪精度较低的现状，提出了一种在 2D 检测中融入深度估计子网络的深度引导 3D 目标检测算法，同时基于 DeepSORT 多目标跟踪框架，采用光流法改进了由于自车运动带来的位置估计不准确问题，并重新设计了外观特征提取网络和外观与运动匹配的策略，经由数据集验证提高了目标遮挡再出现后重匹配效率，并降低了 ID 变换次数。在提高车辆轨迹预测精度方面，在综合考虑了自车，环境与交互行为的基础上，提出了以先预测中点和终点来降低最终轨迹预测误差的车辆轨迹预测算法，在考虑置信度的终点误差与单轨迹预测精度的指标中得到了更好的效果。本文主要研究成果总结如下：

（1）单目 3D 车辆检测

本章搭建的单目 3D 车辆检测模型，首先通过 ResNet50 作为主干网络提取多尺度的基本特征，然后分别建立深度编码器和图像编码器将基本特征转换为对应深度和图像特征。具体而言，在深度编码器中为了更好的估计深度值设计对应的子网络，将由浅及深的每一尺度特征经由多层卷积后进行融合，并不断加深特征的提取，同时将深度估计问题设计为由高斯分布柔和化的分类问题，来削弱因深度估计的不确定性对后续模块的影响，该子网络与后续网络合并后进行监督训练。另一方面，考虑到深度特征可以由车车之间的相对位置进一步学习得到，所以经过 Transformer 进行处理，并与前述估计的深度值作为位置编码相加后得到最终的深度特征。对于图像编码器则是借由 Deformable DETR 的 2D 检测方法，采用多尺度可变形注意力机制对不同尺度的基本特征转换为最终的图像特征。接着在解码阶段，将深度特征作为引导值，与图像特征使用交叉注意力、自注意力和前馈神经网络后得到融合特征。最后通过各检测头输出得到 2D 和 3D 目标的位置，大小，朝向等相关属性。针对所提出的 3D 检测模型在 KITTI 数据集中经过验证和对比，表明在保持较高的 2D 检测精度下有较好的 3D 检测效果。

（2）基于 DeepSORT 的多目标车辆跟踪

本章是基于 DeepSORT 框架进行改进的多目标跟踪算法。该算法主要包含了观测、预测、匹配、更新四个部分，观测是建立在前一章所设计的目标检测基础上，而预测部分是预测上一帧的跟踪目标在当前帧的位置，为了补偿自车运动，在卡尔曼滤波的基础上，采用光流法来估计视角的变换。在匹配阶段，设计了基于 PAN 结构的残差卷积网络来提取外观特征，并结合由马氏距离度量的预测和观测的位置关系进行级联匹配，对于第一次匹配后未能配对的检测目标和跟踪目标再经行第二次的 GIoU 匹配，来实现帧间数据关联。在更新部分中考虑到外观特征的时序关联性，将新特征与旧特征进行线性组合。经过实验表明，改进后的多目标跟踪算法比原本的算法在跟踪精度指标中得到了提高，且 ID 切换数量也相对减少了 19.5%。

（3）车辆轨迹预测

针对车辆轨迹预测问题，需要综合考虑车辆特性、环境约束与车车交互三个方面。车辆特性是对车辆历史轨迹采用 FPN 结构下的因果卷积进行特征提取；对于环境信息则是先将地图重构建成车道中心线的关系拓扑图，并采用空洞车道卷积方式来聚合自节点、左右节点和前后节点的道路特征，环境的约束是将前述特征通过空间交叉注意力机制赋予交通参与者；对于车车交互行为，使用相似的空间自注意力机制完成交互影响的特征传递。借鉴回归终点来限制平均误差的思想，使用残差全连接层分别对中点和多终点进行回归预测。最后，将车辆信息，约束信息，车车交互信息，起点、中点、终点环境信息所融合的特征进行解码，输出最终预测的 6 条轨迹。在 Argoverse 轨迹预测数据集中进行测试，并与其它经典算法进行对比，表明所提出的算法具有更好的预测效果。

（4）实验

将训练好的目标检测以及多目标跟踪算法，应用于实车采集的视频数据并进行定性分析，结果表明所提出的模型能够较稳定的检测并跟踪无人驾驶车辆周围的运动目标。另一方面，轨迹预测算法拓展为在同一场景下的多车辆轨迹预测，通过可视化的结果可以分析出有较好的预测效果。

2. 创新点

（1）为了解决单目 3D 检测的深度不确定性，设计了一个多尺度特征融合的深度估计子网络，同时将该问题设计为高斯分布柔后的分类问题，并进行监督训练。对于 3D 检测网络输出中所包含的估计深度值，除了子网络的输出期望深度结果，还

包括了经由与图像特征融合后解码的深度估计值，将两者取平均后，采用拉普拉斯偶然不确定性的损失函数进行反向传播学习。最终在 2D 检测框架下采用深度特征引导的方式实现了较好的 3D 目标检测任务。

（2）在多目标跟踪中，为了弥补自车运动导致的卡尔曼滤波预测的不确定性，采用了光流法进行运动补偿，同时为了更好的提取车辆的外观特征设计了基于 PAN 结构的残差卷积网络。在外观特征更新中，考虑遮挡的所改变的外观特征以及帧间连贯性问题，将新旧特征进行线性组合。在匹配阶段则是用马氏距离衡量预测的与观测的位置，并将其视为运动相似度与外观相似度进行综合考虑，而非仅仅用距离筛选后的外观相似度进行判断。在第二次匹配中使用 GIoU 来替代原有的 IoU 的交并比进行位置关联。经过上述改进的模块实现了更好的多目标跟踪效果。

（3）在轨迹预测阶段，对于车辆历史轨迹更多的考虑方向和速度信息，同时采用更符合实际的因果卷积来提取历史特征。经过与环境特征融合，车车交互后，对未来轨迹的中点进行预测，来限制因转向导致的轨迹不同程度偏离，然后再对终点位置进行预测，从而限制预测轨迹的平均误差。经过数据集的训练和测试，证明该模型有更好的综合预测表现。

3. 未来工作展望

要搭建完整的不依赖高精地图的纯视觉自动驾驶方案，还有道路识别，可行驶区域识别，视觉里程计的工作要做，碍于个人精力和时间有限，只能在未来进行进一步研究。除此之外，对已提出的算法还能够从如下方面进行研究：

（1）单目的小目标检测还有很大的提升空间，同时如何提高被严重截断和严重遮挡的识别效果也需要更好的数据增强处理。在搭建良好的视觉里程计的前提下是否可以通过多帧来更好的估计深度值。不仅仅是在 DETR 这类 Transformer 的检测框架，还可以尝试搭建更主流的 YOLO 框架下的 3D 目标检测。

（2）在多目标跟踪中，可以采用孪生网络替代卡尔曼滤波的方式进行预测和匹配。除了先检测再跟踪的方法，还可以将检测的特征进一步提取出来进行外观匹配，而不是搭建单独的外观特征网络，可能会有更好的匹配效率。

（3）在轨迹预测中，可以在目标点预测的思想下进一步精细化每个环节所构建的模块，同时对于不同类别的车辆可以分开学习不同的权重参数，而比视为同一种交通参与者的预测可能会有更好的预测效果。

参考文献

[1] 庞松. 科学推动自动驾驶技术发展与应用——拥抱新技术,迎接新挑战[J]. 重庆交通大学学报: 自然科学版, 2021, 40(10): 119-122.

[2] Wang Y, Chao W L, Garg D, et al. Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving[C]. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, 2019: 8437-8445.

[3] Fu H, Gong M M, Wang C H, et al. Deep ordinal regression network for monocular depth estimation[C]. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Salt Lake City, 2018: 2002-2011.

[4] Qi C R, Liu W, Wu C X, et al. Frustum PointNets for 3Dobject detection from RGB-D data[C]. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)， Salt Lake City, 2018: 918-927.

[5] You Y, Wang Y, Chao W L, et al. Pseudo-lidar++: Accurate depth for 3D object detection in autonomous driving[J/OL]. arXiv preprint arXiv: 1906.06310, 2019

[6] Ma X Z, Wang Z H, Li H J, et al. Accurate monocular 3D object detection via color-embedded3D reconstruction for autonomous driving[C]. 2019IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, 2019: 6850-6859.

[7] Y. -N. Chen, H. Dai and Y. Ding. Pseudo-Stereo for Monocular 3D Object Detection in Autonomous Driving[C]. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 2022: 877-887.

[8] Mousavian A, Anguelov D, Flynn J, et al. 3d bounding box estimation using deep learning and geometry[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI, USA, 2017: 7074-7082.

[9] Naiden A, Paunescu V, Kim G, et al. Shift R-CNN: Deep monocular 3d object detection with closed-form geometric constraints[C]. 2019 IEEE International Conference on Image Processing (ICIP). Taipei, Taiwan, 2019: 61-65.

[10] G. Brazil and X. Liu. M3D-RPN: Monocular 3D Region Proposal Network for Object Detection[C]. 2019 IEEE/CVF International Conference on Computer Vision (ICCV),

Seoul, Korea (South), 2019: 9286-9295.

[11] Simonelli A, Bulò S R, Porzi L, et al. Disentangling monocular 3D object detection[C]. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, 2019: 1991-1999.

[12] Qin Z, Wang J, Lu Y. Monogrnet: A geometric reasoning network for monocular 3d object localization[C]. Proceedings of the AAAI Conference on Artificial Intelligence. 2019, 33(01): 8851-8858.

[13] Li P X, Zhao H C，Liu P F, et al. RTM3D: Real-time monocular 3D detection from object keypoints for autonomous driving[C]. Lecture Notes in Computer Science, 2020: 644-660.

[14] Liu Z, Wu Z, Tóth R. SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation [C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Seattle, WA, USA, 2020: 996-997.

[15] Zhou X, Wang D, Krähenbühl P. Objects as points[J/OL]. arXiv preprint arXiv:1904.07850, 2019.

[16] Chen Y J, Tai L, Sun K, et al. MonoPair: Monocular 3D object detection using pairwise spatial relationships[C]. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR). Seattle, 2020: 12090-12099

[17] Xu B, Chen Z Z. Multi-level fusion based 3D object detection from monocular images [C]. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, 2018: 2345-2353.

[18] Zhang Y P Lu J W, Zhou J. Objects are different: Flexible monocular 3D object detection[C]. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, 2021:3288-3297

[19] Ding M Y, Huo Y Q, Yi H W, et al. Learning depth-guided convolutions for monocular 3D object detection[C]. 2020IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, 2020: 11669-11678

[20] Reading C, Harakeh A, Chae J L, et al. Categorical depth distribution network for monocular 3D object detection[C]. 2021 IEEE/CVF Conference on Computer Vision

and Pattern Recognition (CVPR). Nashville, 2021:8551-8560

[21] Breitensiein M, Reichlin F, Leibe B, et al. Online Multi person Tracking by Detection from a Single, Uncalibrated Camera [J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2010, 33(9): 1820-1833.

[22] Xiang Y, Alahi A, Savarese S. Learning to Track: Online Multi-object Tracking by Decision Making[C]. In Proc. of the EEE International Conference on Computer Vision (ICCV). Santiago, Chile, 2015: 4705-4713.

[23] Wang L, Pham N T, Ng T T, et al. Learning deep features for multiple object tracking by using a multi-task learning strategy[C]. 2014 IEEE International Conference on Image Processing (ICIP). Paris, FRANCE, 2014: 838-842

[24] Bewley A, Ge Z, Ott L, et al. Simple Online and Realtime Tracking[C]. In Proc. of the 23rd IEEE International Conference on Image Processing (ICIP). Phoenix, AZ, 2016: 3464-3468.

[25] Wojke N, Bewley A, Paulus D. Simple Online and Realtime Tracking with a Deep Association Metric[C]. 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 2017:3645-3649.

[26] Ullah M, Cheikh F A. Deep Feature Based End-to-End Transportation Network for MultiTarget Tracking[C]. In Proc. of the 25th IEEE International Conference on Image Processing (ICIP). Athens, GREECE, 2018: 3738-3742.

[27] Fang K, Xiang Y, Li X, et al. Recurrent Autoregressive Networks for Online Multi-object Tracking[C]. In Proc. of the 18th IEEE Winter Conference on Applications of Computer Vision (WACV). Lake Tahoe, NV, USA, 2018: 466-475.

[28] Fu Z, Angelini F, Naqvi S M, et al. GM-PHD Filter Based Online Multiple Human Tracking Using Deep Discriminative Correlation Matching[C]. In Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, CANADA, 2018 :4299-4303.

[29] Lefèvre S, Vasquez D, Laugier C. A survey on motion prediction and risk assessment for intelligent vehicles[J]. ROBOMECH journal, 2014, 1(1): 1-14.

[30] Guiggiani M. The science of vehicle dynamics[J]. Pisa, Italy: Springer Netherlands, 2014: 15.

[31] Yi D, Su J, Liu C, et al. Trajectory clustering aided personalized driver intention prediction for intelligent vehicles[J]. IEEE Transactions on Industrial Informatics, 2018, 15(6): 3693-3702.

[32] W. Ding and S. Shen, Online Vehicle Trajectory Prediction using Policy Anticipation Network and Optimization-based Context Reasoning[J/OL]. arXiv preprint arXiv: 1903.00847, 2019.

[33] W. Ding, J. Chen, and S. Shen. Predicting vehicle behaviors over an extended horizon using behavior interaction network[J/OL]. arXiv preprint arXiv: 1903.00848, 2019.

[34] Hu Y, Zhan W, Tomizuka M. Probabilistic prediction of vehicle semantic intention and motion[C]. 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, China, 2018: 307-313.

[35] P. Kumar, M. Perrollaz, S. Lefèvre, and C. Laugier. Learning-based approach for online lane change intention prediction[C]. 2013 IEEE Intelligent Vehicles Symposium (IV). Gold Coast, QLD, Australia, 2013: 797–802.

[36] M. Schreier, V. Willert, and J. Adamy. An integrated approach to maneuver-based trajectory prediction and criticality assessment in arbitrary road environments[J]. IEEE Transactions on Intelligent Transportation Systems, 2016, 10: 2751-2766.

[37] H. Berndt, J. Emmert, and K. Dietmayer. Continuous Driver Intention Recognition with Hidden Markov Models[c]. 2008 11th International IEEE Conference on Intelligent Transportation Systems. Beijing, China, 2008: 1189–1194.

[38] Lefèvre S, Laugier C, Ibañez-Guzmán J. Evaluating risk at road intersections by detecting conflicting intentions[C]. 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. Vilamoura-Algarve, Portugal, 2012: 4841-4846.

[39] Talebpour A, Mahmassani H S, Hamdar S H. Modeling lane-changing behavior in a connected environment: A game theory approach[J]. Transportation Research Procedia, 2015, 7: 420-440.

[40] Liu H X, Xin W, Adam Z, et al. A game theoretical approach for modelling merging and yielding behaviour at freeway on-ramp sections[J]. Transportation and traffic theory, 2007, 3: 197-211.

[41] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory[J]. Neural Computation,

1997, 9(8): 1735-1780.

[42] Zhou G B , Wu J , Zhang C L , et al. Minimal Gated Unit for Recurrent Neural Networks[J]. International Journal of Automation & Computing, 2016, 13(003): 226-234.

[43] W. Ding and S. Shen. Online Vehicle Trajectory Prediction using Policy Anticipation Network and Optimization-based Context Reasoning[J/OL]. arXiv preprint arXiv: 1903.00847, 2019.

[44] S. Dai, L. Li, and Z. Li. Modeling vehicle interactions via modified lstm models for trajectory prediction[J]. IEEE Access, 2019, 7: 38287–38296.

[45] W. Ding, J. Chen, and S. Shen. Predicting vehicle behaviors over an extended horizon using behavior interaction network[C]. Proceedings - IEEE International Conference on Robotics and Automation, 2019: 8634-8640.

[46] L. Xin, P. Wang, C. Chan, J. Chen, S. E. Li, and B. Cheng. Intention aware long horizon trajectory prediction of surrounding vehicles using dual lstm networks[C]. 2018 21st International Conference on Intelligent Transportation Systems (ITSC). Maui, HI, USA, 2018: 1441–1446.

[47] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha. Traffic predict: Trajectory prediction for heterogeneous traffic-agents[J]. In Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33: 6120–6127.

[48] D. Lee, Y. P. Kwon, S. McMains, and J. K. Hedrick. Convolution neural network-based lane change intention prediction of surrounding vehicles for ACC[C]. 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). Yokohama, Kanagawa, Japan, 2017: 1–6.

[49] S. Hoermann, M. Bach, and K. Dietmayer. Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling[C]. 2018 IEEE International Conference on Robotics and Automation (ICRA). Brisbane, QLD, Australia, 2018: 2056–2063.

[50] N. Deo and M. M. Trivedi. Convolutional social pooling for vehicle trajectory prediction[C]. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 2018: 1549-15498.

[51] M. Schreiber, S. Hoermann, and K. Dietmayer. Long-term occupancy grid prediction using recurrent neural networks[C]. Proceedings - IEEE International Conference on Robotics and Automation, 2019: 9299-9305.

[52] Hou L, Xin L, Li S E, et al. Interactive trajectory prediction of surrounding road users for autonomous driving using structural-LSTM network[J]. IEEE Transactions on Intelligent Transportation Systems, 2019, 21(11): 4615-4625

[53] R. Chandra, et al. Forecasting trajectory and behavior of road-agents using spectral clustering in graph-LSTMS[J]. IEEE Robotics and Automation Letters, 2020, 5(3): 4882–4890.

[54] M.-F. Chang et al. Argoverse: 3D Tracking and Forecasting With Rich Maps[C]. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019: 8748–8757.

[55] J. Gao, S Chen, et al. VectorNet: Encoding HD Maps and Agent Dynamics From Vectorized Representation[C]. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA, 2020: 11525–11533.

[56] M. Liang et al. Learning Lane Graph Representations for Motion Forecasting[J]. i Computer Vision – ECCV 2020, 2020: 541-556.

[57] H. Zhao et al. TNT: Target-driveN Trajectory Prediction[J/OL]. arXiv preprint arXiv: 2008.08294, 2020.

[58] J. Gu, C. Sun, and H. Zhao. DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets[C]. 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, QC, Canada, 2021: 15303–15312.

[59] Thomas Gilles, Stefano Sabatini, et al. HOME: Heatmap Output for future Motion Estimation[J/OL]. arXiv preprint arXiv:2105.10968, 2021.

[60] K. He, X. Zhang, S. Ren and J. Sun. Deep Residual Learning for Image Recognition[C]. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778.

[61] T. -Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie. Feature Pyramid Networks for Object Detection[C]. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 936-944.

[62] Vaswani A ,Shazeer N ,Parmar N , et al. Attention Is All You Need[J/OL]. arXiv preprint arXiv: 1706.03762, 2017.

[63] Carion N, Massa F, Synnaeve G, et al. End-to-End Object Detection with Transformers [J/OL]. arXiv preprint arXiv:2005.12872, 2020.

[64] Zhu X , Su W , Lu L , et al. Deformable DETR: Deformable Transformers for End-to-End Object Detection[J/OL]. arXiv preprint arXiv:2010.04159, 2021.

[65] T. -Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár. Focal Loss for Dense Object Detection[C]. 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017: 2999-3007.

[66] Z. Tian, C. Shen, H. Chen and T. He, FCOS: Fully Convolutional One-Stage Object Detection[C]. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South), 2019: 9626-9635.

[67] Y. Chen, L. Tai, K. Sun and M. Li, MonoPair: Monocular 3D Object Detection Using Pairwise Spatial Relationships[C]. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA, 2020: 12090-12099.

[68] Kingma D P, Ba J. Adam: A method for stochastic optimization[J/OL]. arXiv preprint arXiv:1412.6980, 2014.

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

[70] Xiaozhi Chen, Kaustav Kundu, et al. 3d object proposals for accurate object class detection[C]. International Conference on Neural Information Processing Systems. MIT Press, 2015.

[71] Chen X, Kundu K, Zhang Z, et al. Monocular 3d object detection for autonomous driving [C]. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2016: 2147–2156.

[72] Ma X, Zhang Y, Xu D, et al. Delving into localization errors for monocular 3d object detection [C]. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021: 4721–4730.

[73] R Zhang, H Qiu, et al. MonoDETR: Depth-guided Transformer for Monocular 3D Object Detection[J/OL]. arXiv preprint arXiv:2203.13310, 2022.

[74] C Shu, Z Chen, et al. SideRT: A Real-time Pure Transformer Architecture for Single Image Depth Estimation[J/OL]. arXiv preprint arXiv: 2204.13892, 2022.

[75] Lucas B D, Kanade T. Schunck, Determining optical flow[J]. Artificial Intelligence, 1981, 17(1-3), pp.185-203.

[76] S Liu, L Qi, et al. Path Aggregation Network for Instance Segmentation[J/OL]. arXiv preprint arXiv: 1803.01534, 2018

[77] 刘鑫辰.城市视频监控网络中车辆搜索关键技术研究[D].北京邮电大学, 2018.

[78] W Zeng, M Liang, et al. LaneRCNN: Distributed Representations for Graph-Centric Motion Forecasting[J/OL]. arXiv preprint arXiv: 2101.06653, 2021.

[79] Y Liu, J Zhang, et al. Multimodal Motion Prediction with Stacked Transformers[J/OL]. arXiv preprint arXiv: 2103.11624, 2021.

[80] Thomas Gilles, Stefano Sabatini, et al. GOHOME: Graph-Oriented Heatmap Output for future Motion Estimation[J/OL]. arXiv preprint arXiv:2109.01827, 2021.

攻读学位期间发表论文与研究成果清单

学术论文：

[1] Hongbin Ren, **Gaoli Zhou**, Hongwei Zhang, et al. Interaction-Based Trajectory Prediction of Surrounding Vehicles with Driving Maneuvers Recognition[C]. Proceedings of 2022 International Conference on Autonomous Unmanned Systems (ICAUS). Springer, Singapore, 2023, 1010: 777-790. (EI 检索)

[2] Hongbin Ren, Jiyu Sun, **Gaoli Zhou**, et al. Path Planning Method for Automatic Parking Based on Hybrid A*[C]. Proceedings of 2022 International Conference on Autonomous Unmanned Systems (ICAUS). Springer, Singapore, 2023, 1010: 763-776. (EI 检索)

[3] 任宏斌,**周高立**.基于凸优化的自动泊车轨迹规划方法[C]. (SAECCE 已接受)

参研项目：

[1] 2020.9-2021.6 参与“跨越险阻 2020”，负责高精地图绘制、路径规划及现场调试工作。

致谢

三年硕士生活如同春雨，起初我撑着一把旧伞，害怕要承受的是滂沱与怒号，当慢慢地伸出手去触碰它时，却是别样的清凉和温柔，当自负的以为捧住了整个春天，却又只是淋湿了袖口；渐渐的，我学会了去观察躲在屋檐下的燕子，去欣赏倒影里摇曳的枝柳，去寻找在雨中绽放的白花；可时间一长，雨大了起来，多了些寒意，鞋子也被浸湿了，打了个喷嚏后不由自主的感到无助与担心，当发现路上还有零星的行人在前往自己的目的地时，不禁自我怀疑，我将去向何方；现在雨渐停，暗自懊恼，以为错过了春天，却发现四周竟是那么生机勃勃，原来一切才刚刚开始。从无知的我踏入校园开始，能以充实的头脑心满意足地迎接下一阶段的生活，便是这段岁月我最大的收获。

由衷地感谢任宏斌导师对我的接纳与包容，在学术上，任老师总是积极的鼓励学生去钻研探索，并以严谨的治学态度教导我们认真对待学术知识，遇到不解时又能用渊博的知识为我们指点迷津，开拓思路；在生活上，我不幸的经历了新冠、甲流和复阳三次严重高烧，任老师也总是亲切关怀我的健康状况并给予了及时的帮助，除此之外还会带学生们聚餐，团结学习氛围等等，让我感受到学习也可以是如此自由与快乐。

除此之外还要衷心地感谢实验室里风趣幽默的吴志成老师、和善可敬的赵玉壮老师、诲人不倦的齐志权老师、乐于助人的杨林老师以及尽职尽责的张斌老师，感谢老师们对我学习和生活中提供的莫大帮助。感谢实验室的同学，在与你们的每次组会以及日常的交流中，我总能有新的收获，与你们共同进步的日子也是如此快乐。此外感谢朱晶，吴世南，王浩，谢盼盼，张洪伟等在我论文写作中提供的帮助。感谢挚友孙静，徐方栋，王勇，刘晓彤等在科研与日常生活对我的帮助，与你们一起游玩旅行让我看到了更广阔的世界。

感谢父母对我求学生涯的默默支持，在这二十多年来对我无理由的信任与鼓励，这也是我前进的最大动力。

最后感谢百忙之中为论文审阅及答辩付出劳动的各位专家老师。

周高立

2023 年 6 月