第 48 卷 第 21 期/2021 年 11 月/中国激光 研究论文

中国激光

融合附加神经网络的激光雷达点云单目标跟踪

周笑宇[1]，王玲[1*]，马燕新[2]，陈沛铂[1]

[1]国防科技大学电子科学学院，湖南 长沙 410073；

[2]国防科技大学气象海洋学院，湖南 长沙 410073

摘要 已有激光雷达点云单目标跟踪工作对分布稀疏、小规模点云目标的跟踪性能不佳。针对该问题，提出了一种融合附加神经网络的点云单目标跟踪算法。所提算法在网络训练时，利用附加网络执行前景点云分割和中心坐标偏移回归的附加任务，引导骨干网络学习稀疏、小规模目标点云的高鉴别力特征，供后续网络在逐帧点云中完成对目标的定位和与背景的区分；网络推断时，则绕过附加网络，直接由骨干网络提取目标点云特征，在保证跟踪任务准确性的同时满足任务实时性的要求。在 KITTI 数据集上的测试结果表明：相较于已有工作，所提算法在相同参数设定下平均跟踪成功率提高了 0.89 个百分点，平均跟踪准确率提高了 2.51 个百分点；并且通过对参数设定的进一步研究与调整，最终所提算法平均跟踪成功率提高了 4.54 个百分点，平均跟踪准确率提高了 7.83 个百分点，且对分布稀疏、点数较少的点云目标有明显的性能提升。

关键词 图像处理；激光雷达；点云；附加网络；单目标跟踪

中图分类号 TN249 **文献标志码** A **doi**：10.3788/CJL202148.2110001

1 引 言

激光雷达通过发射扫描激光信号，并将从物体反射的信号转化为点云数据，记录三维空间中物体的表面形状、空间位置等信息。点云数据为三维场景下处理目标分类[1-3]、检测[4]、分割[5]、重建[6]和跟踪[7-9]等任务提供了理想的数据形式。以单目标跟踪技术为例，相较于目前主流的二维视频或图像序列，点云数据在作为数据方案时，主要有以下两个优势：1）点云可以更好描述真实场景中目标的三维空间几何信息，比如目标的位置、尺度和姿态等；2）不同于摄像头的被动光学成像原理，激光雷达以主动成像的方式采集信息，不受自然光条件的影响，使得点云数据对夜晚、雨雪等不同复杂环境具有适应性，对眩光、反光和阴影等不同状况具有鲁棒性。因此，基于激光雷达点云的单目标跟踪任务具有较高的研究价值和实用意义。

单目标跟踪任务侧重解决单个未知类别的通用目标跟踪问题。以代表性的基于二维图像序列或视频序列的视觉单目标跟踪任务为例，基本任务要求是利用给定的初始帧信息，确定需要跟踪的目标，并在后续各帧中预测二维边界框，对目标进行定位。在整个过程中，该目标实例的真值信息只有在跟踪开始时才能从初始帧真值信息中确定。单目标跟踪的通用技术框架[10]包括运动模型、目标模型、观察模型、更新模型 4 部分。其中运动模型基于对前一帧的估计结果，生成一组候选区域或边界框，这些区域或边界框可能包含当前帧中的目标；目标模型通过设计特征提取器，使用一些特征表示方法为目标和候选集中的对象建立表征模型；观察模型根据从候选对象中提取的特征来判断该候选对象是否为目标，实际可视为一个目标背景二分类器；更新模型控制更新观察模型的策略和频率，在适应目标变化和跟踪漂移之间取得平衡。

相较于二维视觉单目标跟踪领域，目前三维点云单目标跟踪领域成果仍然较少，但已有工作对其他领域的优秀思想与方法进行了有效学习与借鉴，并结合点云数据的特性进行了改进，取得了较好的成果。近年来，孪生型（Siamese）框架逐步成为了二维视觉单目标跟踪领域的一大研究主流，Giancola 等[8]基于此提出了第一个应用于点云的三维孪生型

收稿日期：2021-02-22；**修回日期**：2021-03-28；**录用日期**：2021-04-15

通信作者：*wangling@nudt.edu.cn

网络跟踪器。这个 Siamese 结构的骨干网络基于 Achlioptas 等[11]的工作，以一个自编码器网络在 Siamese 结构的两条支路上分别进行模板点云和候选点云的特征学习，将它们编码为紧凑的隐层表征向量，而后采用余弦相似度算法来计算模板向量和候选向量之间的相似度，从而确定下一帧中被跟踪对象的位置。但是，该跟踪器使用穷举法搜索采样的候选对象，来对模板点云之间进行相似性度量，这种搜索策略保证每帧都包含真值边界框，却无视框本身可能就包含真值的提案区域的情况，没有全面分析连续搜索空间。Zarzar 等[7]在此基础上加入了点云鸟瞰图作为信息源，利用视觉跟踪算法 SiamRPN[12]选择提案区域，降低了确定目标点云搜索空间时的计算复杂度，使实时性能得到了提升。Zou 等[13]在 SC3D 算法[2]工作的基础上，提出了一种在线准确性验证方法。该方法通过融合 2D 图像和 3D 点云来利用各种信息，以显著减轻串行网络结构对 2D 跟踪结果的依赖性并缩小 3D 搜索空间，将 3D 搜索空间直接输入到现有 3D 跟踪器的骨干网络中，取得了明显的性能提升。Fang 等[14]借鉴了 SiamRPN[12]的思想，设计了 3D-SiamRPN 算法，该算法通过区域提案网络中的互相关模块得到预测的目标边界框，在实时性和准确性之间取得了较好的平衡。Qi 等[9]则借鉴点云目标检测领域先进工作 VoteNet[15]的思想，提出了 P2B 算法。首先，该算法以 PointNet++[16]作为主干网络，在 Siamese 网络架构两支路上分别将第一帧目标模板点云和当前帧搜索区域点云中大量表面局部信息聚合到少量的种子点上；其次，通过设计的特征增强模块，将目标特征信息融合到搜索区域信息中；然后，通过对搜索区域各种子点进行霍夫投票和分类，得到了潜在的目标中心点；最后，基于这些中心点，又通过聚类操作回归得到目标的提案框和对应的目标得分，最终选出预测结果。该工作较好地解决了 SC3D[8]工作中遇到的确定候选区域计算消耗大、时间长的问题，取得了准确率和实时性的提升。但是，该算法在第一帧目标点云点数较少、分布稀疏的情况下跟踪效果不佳，容易出现跟踪漂移，鲁棒性和网络对稀疏小样本点云的特征提取能力有待提升。

针对上述问题，本文在 P2B 算法的基础上提出一种融合附加网络的点云单目标跟踪算法。该算法借鉴点云目标检测工作 SA-SSD 算法[17]的思想，引入附加网络执行附加任务，辅助引导骨干网络学习更细粒化的稀疏小样本点云特征，在提高网络对此类点云跟踪成功率的同时，满足跟踪任务的实时性要求。为验证所提算法的效果，使用 KITTI 跟踪数据集[18]中的场景点云序列测试了所提算法性能。

2 P2B 跟踪算法

P2B 跟踪算法主要步骤如下。1）目标特征提取与增强：通过一个两分支权重共享的孪生网络，分别提取第一帧目标模板点云和后续搜索区域点云的特征，然后再将这些特征统一输入到特征增强网络，计算目标模板点云和搜索区域点云特征之间的相似度分布图，并与目标专属特征进行特征融合，达到特征表征能力增强的目的。2）三维目标边界框提案的生成和选择：主要的思想是借助特征点进行霍夫投票、聚类，回归出可能的目标中心和候选边界框提案，以提案的目标性得分作为依据预测出最终的目标边界框。

2.1 网络结构

图 1 为 P2B 跟踪算法的网络结构示意图。

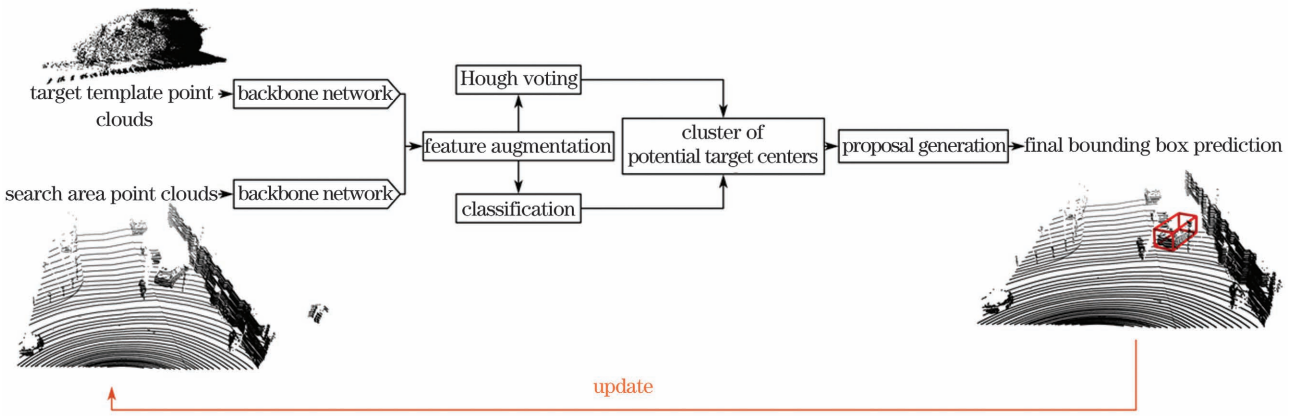

图 1 P2B 算法网络结构示意图

Fig. 1 Structure schematic diagram of P2B algorithm network

目标模板点云和搜索区域点云分别输入到孪生网络两个分支的骨干网络中。P2B 算法选择 PointNet++作为骨干网络，对点云进行降采样，并将点云逐点的局部领域特征聚合到了降采样后的种子点中。图 2 为特征增强部分的流程示意图，其中 MLP_1 和 MLP_2 均为多层卷积层（Conv2d 或 Conv1d）、批归一化层（BatchNorm2d）和激活函数层（ReLU）的组合神经网络。对模板点云种子点的坐标及特征（对应图 2 中 template seeds and feature）进行复制（对应图 2 中 copy），再与余弦相似度分布图进行拼接（对应图 2 中 concatenate），最后经过多层神经网络和最大值池化操作（对应图 2 中 Maxpooling）的处理，实现了目标专属特征的融合与增强，充分将目标专属信息应用到后续的霍夫投票（对应图 1 中 Hough voting）、计算分类得分（对应图 1 中 classification）中。

图 3 为提案生成部分示意图，其中分类（对应图

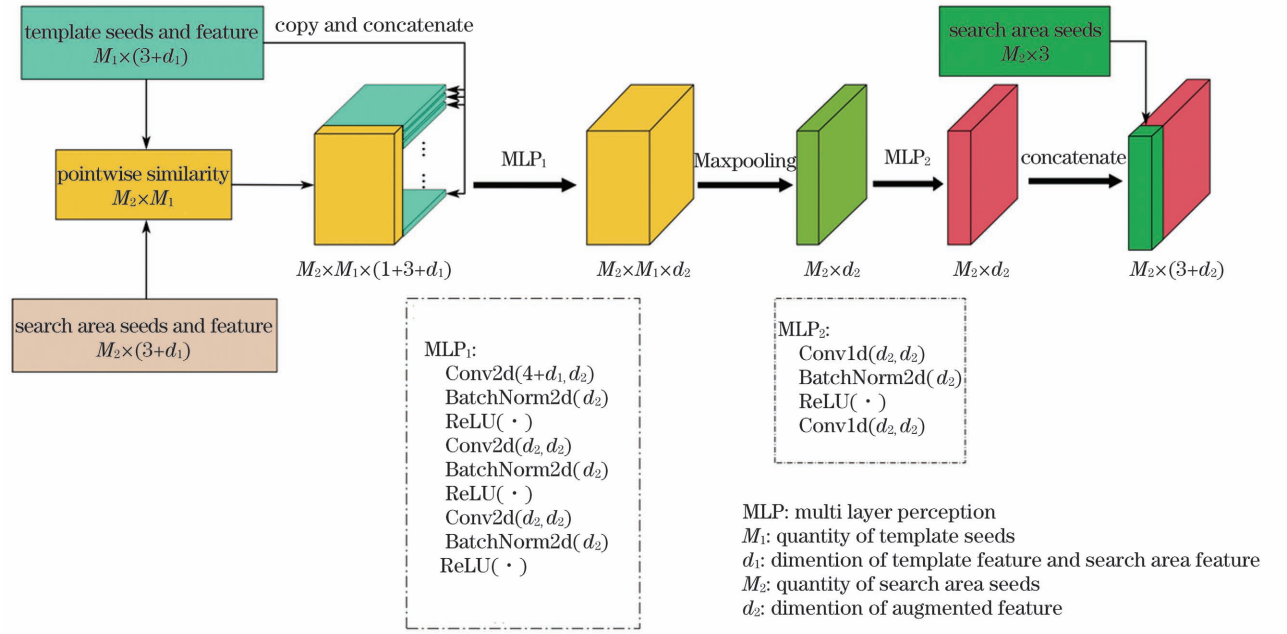

图 2 特征增强部分示意图

Fig. 2 Schematic diagram of feature augmentation

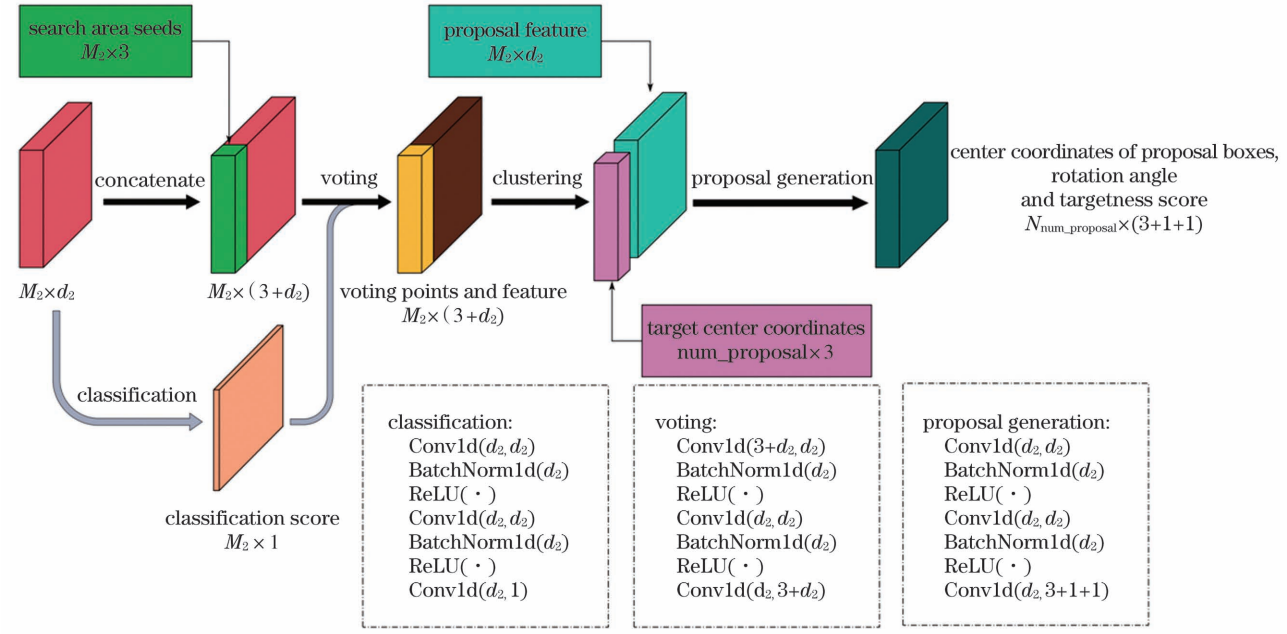

图 3 提案生成部分示意图

Fig. 3 Schematic diagram of proposal generation

中 classification)、霍夫投票(对应图中 voting)和提案生成(对应图中 proposal generation)均由 Conv2d 或 Conv1d、BatchNorm2d 和 ReLU 的组合网络实现,而潜在目标中心点的聚类(对应图中 clustering)是借鉴 PointNet++[16]的 set abstraction 层,通过聚类、球查询和特征融合,得到潜在的目标中心点坐标(对应图中 target center coordinates)和提案特征(对应图中 proposal feature)实现的。生成总数为 $N_{\text{num\_proposal}}$ 的提案,每个提案包含了目标边界框的中心坐标(对应图中 center coordinates of proposal boxes)、旋转偏角(对应图中 rotation angle)和目标性得分(对应图中 targetness score)。以最大的目标性得分为依据,选择某个提案边界框作为网络在当前帧点云对目标的预测。最后,预测边界框,在下一帧中更新搜索区域点云,继续进行跟踪任务。

2.2 损失函数

P2B 算法的损失函数主要由 4 部分组成。第 1 部分是搜索区域种子点与目标中心点坐标偏移的回归损失 L_{REG},表达式为

$$L_{\text{REG}}=\frac{1}{M}\sum_{j}\|\Delta\boldsymbol{x}_j-\Delta\boldsymbol{gt}_j\|\cdot I(s_j),\tag{1}$$

式中:M 表示种子点数量;$\Delta\boldsymbol{x}_j$ 表示第 j 个种子点 s_j 与目标中心点之间的实际坐标偏差;$\Delta\boldsymbol{gt}_j$ 表示种子点 s_j 与目标中心点之间的真值坐标偏差;$I(s_j)$ 表示种子点 s_j 是否在目标表面上,即

$$I(s_j)=\begin{cases}1, & s_j \text{ on target's surface}\\ 0, & \text{else}\end{cases},\tag{2}$$

则只将在目标表面上的种子点用于训练和损失函数的计算。

第 2 部分是搜索区域种子点目标背景二分类损失 L_{cla},即将位于目标表面的种子点作为正样本,其余的作为负样本,应用标准二维交叉熵损失函数来计算搜索区域种子点分类产生的损失,表达式为

$$L_{\text{CLA}}=\{l_1,l_2,\cdots,l_I\},\tag{3}$$

$$l_i=-\{a\cdot s_i^{(\text{s})}\cdot\log_2\sigma(S_i)+(1-S_i)\times\log_2[1-\sigma(s_i^{(\text{s})})]\},\ i=1,2,\cdots,I,\tag{4}$$

式中:a 为正样本的权重;$s_i^{(\text{s})}$ 为第 i 个搜索区域种子点的预测类别标签;S_i 为第 i 个搜索区域种子点的真值类别标签;$\sigma(\cdot)$为 Sigmoid 函数。

第 3 部分是提案框目标性得分损失 L_{PROP},即将中心到目标中心的距离小于一定阈值的提案作为正样本,其余的提案作为负样本,应用标准二维交叉熵损失函数来计算提案框分类产生的损失,表达式为

$$L_{\text{PROP}}=\frac{\sum_{n=1}^{N_{\text{num\_proposal}}}w_n l_n}{\sum_{n=1}^{N_{\text{num\_proposal}}}P_n},\tag{5}$$

$$l_n=-\{a\cdot s_n^{(\text{p})}\cdot\log_2\sigma(P_n)+(1-P_n)\times\log_2[1-\sigma(s_n^{(\text{p})})]\},\ n=1,2,\cdots,N_{\text{num\_proposal}},\tag{6}$$

式中:$N_{\text{num\_proposal}}$ 为候选提案数量;P_n 为第 n 个候选提案的真值标签;$s_n^{(\text{p})}$ 为第 n 个候选提案的目标分类得分;w_n 是由正负样本确定的权重,使得到目标中心偏移较小或较大的候选提案都能在计算损失时发挥作用。

第 4 部分是提案框坐标及旋转量回归损失 L_{BOX},即应用 SmoothL1 损失函数[19]对正样本提案边界框的参数计算损失,表达式为

$$L_{\text{BOX}}=\begin{cases}\dfrac{\sum_{n=1}^{N_{\text{num\_proposal}}}P_n\dfrac{0.5}{\beta}(\boldsymbol{B}_{\text{E}n}-\boldsymbol{B}_{\text{G}n})^2}{\sum_{n=1}^{N_{\text{num\_proposal}}}P_n}, & \|\boldsymbol{B}_{\text{E}n}-\boldsymbol{B}_{\text{G}n}\|<1\\ \dfrac{\sum_{n=1}^{N_{\text{num\_proposal}}}P_n(\|\boldsymbol{B}_{\text{E}n}-\boldsymbol{B}_{\text{G}n}\|-0.5\beta)}{\sum_{n=1}^{N_{\text{num\_proposal}}}P_n}, & \text{else}\end{cases},\tag{7}$$

式中:β 为参数;$\boldsymbol{B}_{\text{E}n}$ 为第 n 个候选提案框的坐标及旋转量参数;$\boldsymbol{B}_{\text{G}n}$ 为目标真值边界框的坐标及旋转量参数。

最终的网络训练损失函数可表达为

$$L=\gamma_1 L_{\text{CLA}}+\gamma_2 L_{\text{PROP}}+\gamma_3 L_{\text{BOX}}+\gamma_4 L_{\text{REG}},\tag{8}$$

式中:γ_1、γ_2、γ_3、γ_4 均为参数。

3 所提算法

3.1 网络结构

所提算法网络结构如图 4 所示。骨干网络在提取特征的同时,对点云也进行了逐层降采样、邻域特征聚合等操作,无疑会损失目标点云的细节特征信息。但在三维点云的其他任务,如目标分割中,更加追求逐点级别上的特征学习,以实现逐点分类,实际上对于目标整体的特征表示更加细致,相应的特征提取模块更能适应小目标稀疏点云的特征学习。因此所提算法网络在 P2B 算法网络的基础上,在网络训练阶段加入了附加网络部分,通过执行附加的点

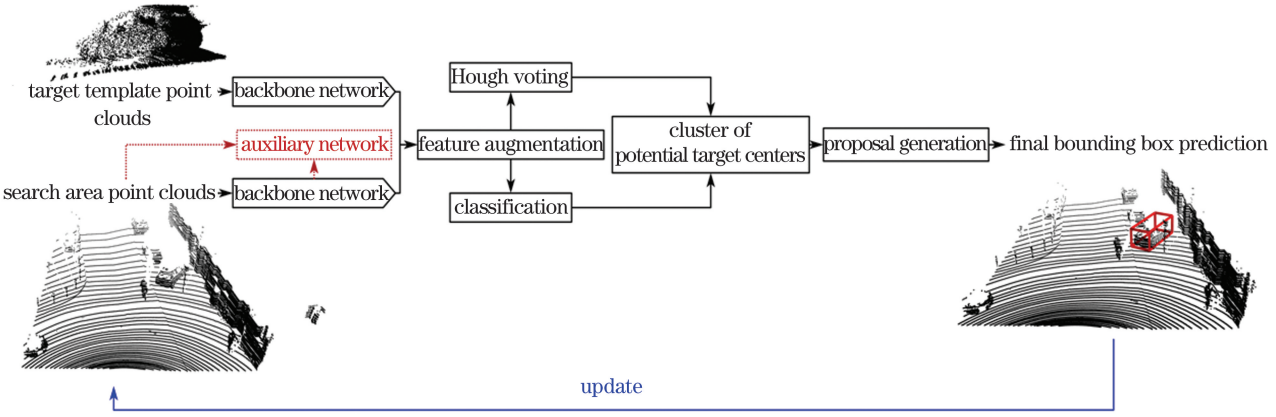

图 4 所提算法网络结构示意图

Fig. 4 Structure schematic diagram of proposed algorithm network

云前景分割任务和逐点中心坐标偏移回归任务，将附加任务损失函数同跟踪任务损失函数一起用于优化骨干网络和附加网络的网络参数。事实上，所提算法在原算法上添加的这部分网络，可以在网络推断阶段继续保留在整体网络结构中，实现跟踪、分割的多任务学习。但是，由于跟踪任务在推断阶段的实时性，这部分网络只适合在训练阶段作为附加网络，而在推断阶段需要绕过该网络，专注于执行主要的跟踪任务。

3.2 附加网络及损失函数设计

所提算法网络训练阶段流程如图 5 所示。

1）将目标模板点云和搜索区域点云分别输入到骨干网络中，提取特征和种子点输入到特征增强和提案生成网络（对应图中 feature augmentation & proposal generation）中，输出若干候选提案框的坐标、旋转角和目标性得分及搜索区域种子点分类得分。

2）将骨干网络提取的搜索区域点云特征和种子点输入到附加网络中。这部分网络由3部分组

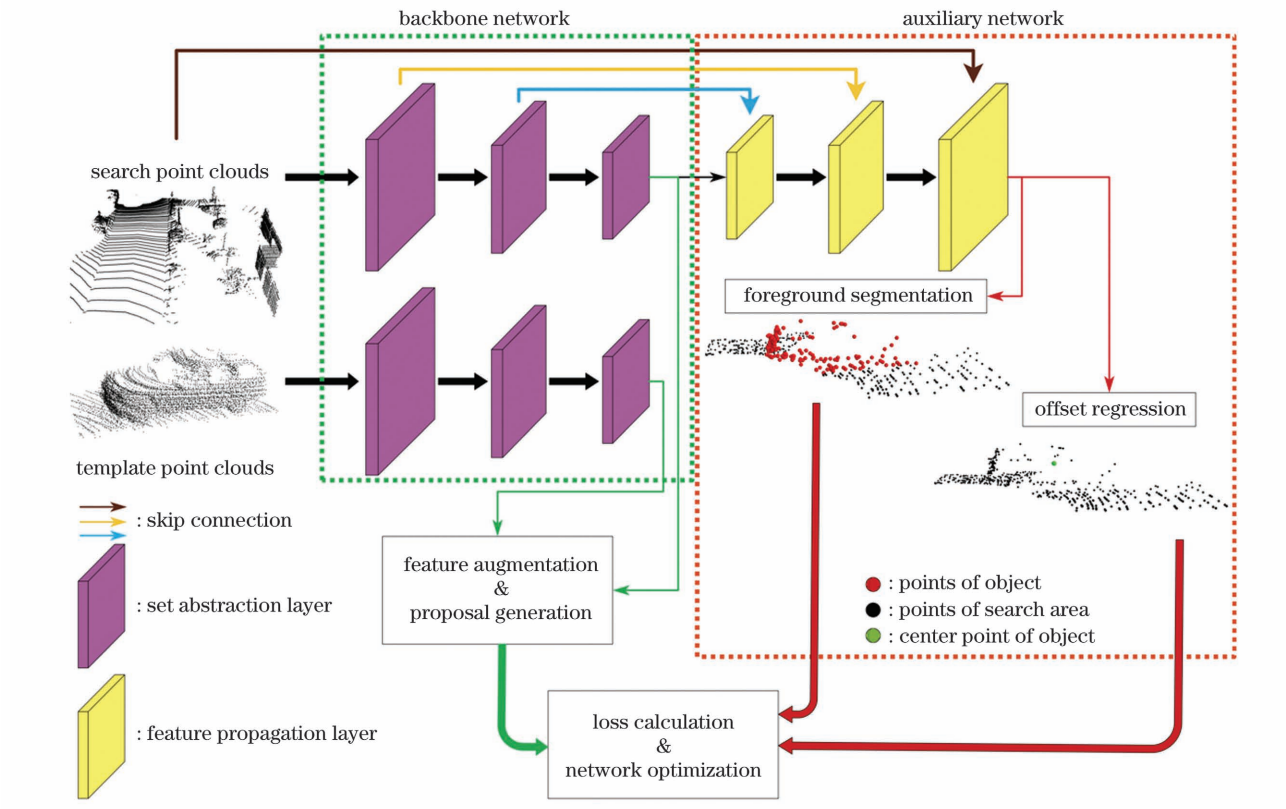

图 5 网络训练阶段流程图

Fig. 5 Flow chart of network training

成。第 1 部分是附加特征提取模块，由多个特征传播层[16]（对应图中 feature propagation layer）组成，作用是与骨干网络的跨层链接、邻域特征加权聚合和卷积设计，使得骨干网络中多层、不同分辨率的目标特征最终转化为逐点特征，作为后续附加任务的基础。特征传播层主要由最近邻层、加权插值层（3-NN interpolation）、Conv2d、BatchNorm2d 和 ReLU 组成，如图 6 所示。其中最近邻层确定各点的三个最近邻种子点的索引，而 3-NN interpolation 则根据各点的近邻索引，将对应的三个种子点的特征加权求和，作为该点的新特征。权值的计算方式为

图 6　特征传播层网络结构示意图

Fig. 6　Structure schematic diagram of feature propagation layer

$$w_m(\boldsymbol{p}_k)=\begin{cases}\dfrac{1}{\|\boldsymbol{p}_k-\boldsymbol{p}_m\|_2}, \boldsymbol{p}_m\in N(\boldsymbol{p}_k)\\ 0, \text{otherwise}\end{cases},$$
$$k=1,2,\cdots,K,\ m=1,2,3, \tag{9}$$

式中：$N(\boldsymbol{p}_k)$为点 $\boldsymbol{p}_k$ 的邻域；K 为该层对应的采样点总数。Conv2d、BatchNorm2d 和 ReLU 则对邻域特征进行进一步的提炼。第 2 部分是附加回归任务模块，作用是回归逐点与目标中心点的坐标偏移信息，该模块主要由全连接层组成。第 3 部分是附加分割模块，作用是逐点进行前景目标和背景分类，该模块主要由全连接层组成。

3）将前 2 步的输出量输入到损失函数中，计算各部分的损失并加权求和，然后反向传播，对网络参数进行优化（对应图 5 中 loss calculation & network optimization）。

附加网络执行的两项附加任务引入了两部分损失。第 1 部分是前景分割结果和逐点真值分类标签之间的损失 L_{SEG}，应用 Sigmoid focal loss 函数[20]进行计算，表达式为

$$L_{\text{SEG}}=-\{S_{\text{EST}}\cdot\log_2\sigma(S_{\text{GT}})+(1-S_{\text{GT}})\cdot\log_2[1-\sigma(S_{\text{EST}})]\}\cdot W_{\text{FOCAL}}, \tag{10}$$
$$W_{\text{FOCAL}}=\{[1-\sigma(S_{\text{EST}})]\cdot S_{\text{GT}}+\sigma(S_{\text{EST}})\cdot[1-\sigma(S_{\text{GT}})]\}^{\gamma}\cdot W_0, \tag{11}$$
$$W_0=\alpha\cdot S_{\text{GT}}+(1-\alpha)\cdot(1-S_{\text{GT}}), \tag{12}$$

式中：α 和 γ 均为参数；S_{GT} 为真值分割标签；S_{EST} 为预测分割标签。

第 2 部分是偏移量回归损失 L_{OFFSET}，即搜索区域点云中各点与目标中心之间坐标偏移量的回归损失，应用 SmoothL1 损失函数[19]进行计算，表达式为

$$L_{\text{OFFSET}}=\begin{cases}\dfrac{0.5}{\beta}(\boldsymbol{O}_{\text{EST}}-\boldsymbol{O}_{\text{GT}})^2, \|\boldsymbol{O}_{\text{EST}}-\boldsymbol{O}_{\text{GT}}\|<1\\ \|\boldsymbol{O}_{\text{EST}}-\boldsymbol{O}_{\text{GT}}\|-0.5\beta, \text{else}\end{cases}, \tag{13}$$

式中：$\boldsymbol{O}_{\text{EST}}$ 为预测坐标偏移量；$\boldsymbol{O}_{\text{GT}}$ 为真值坐标偏移量。

综上所述，所提算法的损失函数表达式为

$$L_{\text{NEW}}=\gamma_1 L_{\text{CLA}}+\gamma_2 L_{\text{PROP}}+\gamma_3 L_{\text{BOX}}+\gamma_4 L_{\text{REG}}+\gamma_5 L_{\text{SEG}}+\gamma_6 L_{\text{OFFSET}}, \tag{14}$$

式中：γ_1、γ_2、γ_3、γ_4、γ_5、γ_6 均为参数。

4　实验结果分析

4.1　实验环境及参数设定

所提算法运行平台配置：CPU 为 Intel i7 8750H，内存为 32 GB，显卡为 Nvidia RTX2080Ti，操作系统为 64 位 Windows10，所采用的编程环境为 Python3.6，深度学习框架为 Pytorch1.3.0。算法参数设定如下：损失函数的参数分别设定为 $\gamma_1=0.2$，$\gamma_2=1.5$，$\gamma_3=0.2$，$\gamma_4=1$（参照文献[9]进行设定）和 $\gamma_5=0.9$，$\gamma_6=2$（参照文献[17]进行设定）；目标模板点云输入点数为 512，种子点数为 64；搜索区域点云输入点数为 1024，种子点数为 128；候选提案数目 $N_{\text{num\_proposal}}$ 等于 64（参照文献[9]进行设定）；搜索区域种子点目标背景二分类损失 L_{cla} 的正样本的权重 $a=1$；前景分割结果和逐点真值分类标签之间的损失 L_{SEG} 的参数 $\alpha=0.25$，$\gamma=2$（参照文献[9]进行设定）；偏移量回归损失 L_{OFFSET} 的参数参照文献[9]进行设定；网络学习率 $\eta=0.001$；小批量学习数据大小 batchsize 等于 32；训练轮数为 36。

4.2　利用 KITTI 数据集测试算法性能

对 KITTI 跟踪数据集[18]的训练集进行划分，作为训练和测试算法性能的数据集，其中包含了超过 20000 个在激光雷达（HDL-64E，Velodyne）采集的场景点云中人工标注的点云目标。具体的划分方法为：17 个序列（0～16）作为训练集，2 个序列（17～18）作为验证集，2 个序列（19～20）作为测试集。此

划分方式与已有工作 SC3D[8] 和 P2B[9] 保持一致，有利于对比不同算法之间的性能。

根据单目标跟踪的一次评估(OPE)思想[21]，选择成功率和准确率作为评估指标，其中成功率(R_S)为预测框和真值边界框之间的交并比(IOU)，准确率(R_P)为误差(预测框和真值框中心之间的距离)在 0～2 m 范围内的曲线下面积(AUC)，即误差距离分布直方图积分的结果。

表 1 详细对比了 SC3D[8]、P2B[9] 和所提算法对 KITTI 数据集[18] 中不同类别目标的跟踪性能，其中表内第一行数字代表测试集中各类别目标的标注数量。从表中可以看出，所提算法在类别 Car、Van 数据上的跟踪性能较 P2B 算法均有提升，在初始模板点云数量较少的 Pedestrian 类数据上，跟踪准确率有明显的性能提升，说明加入的附加网络模块对于小规模模板点云情况下的目标特征有效学习有促进作用。

部分跟踪结果如图 7～10 所示，其中深色边界框为真值边界框，浅色边界框为算法预测边界框，黑点为搜索区域点云。同时，增加的附加网络执行的逐点分割任务结果如图 11～14 所示，其中上面一行为算法预测目标点，下面一行为真值目标点。

表 1 不同算法对各类目标的跟踪性能对比

Table 1 Comparison of tracking performance of different algorithms for various targets unit: %

Parameter	Method	Car 6424	Pedestrian 6088	Cyclist 308	Van 1248	Mean 14068
R_S	SC3D	41.30	18.20	41.50	40.40	31.20
	P2B	56.20	28.70	32.10	40.80	42.40
	Proposed algorithm	56.98	28.40	30.44	48.60	43.29
R_P	SC3D	57.90	37.80	70.40	47.00	48.50
	P2B	72.80	49.60	44.70	48.40	60.00
	Proposed algorithm	73.73	52.40	41.67	59.25	62.51

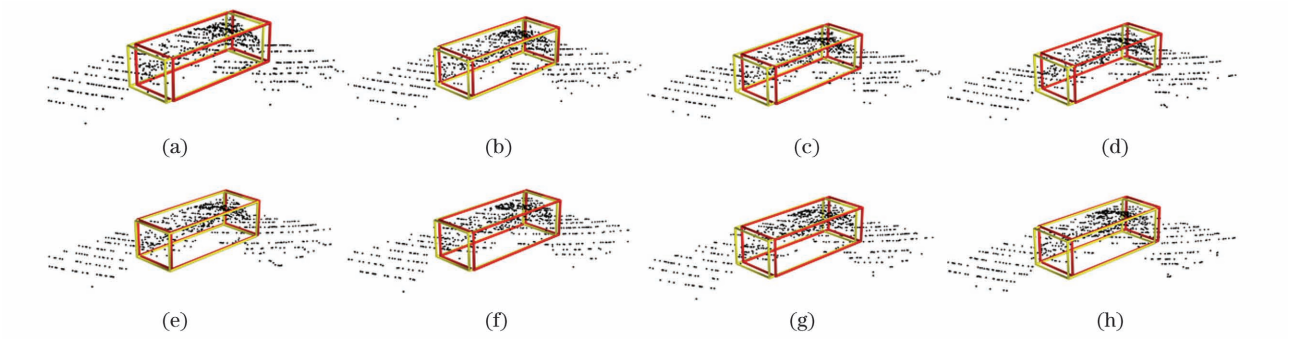

图 7 Car 类目标部分跟踪结果。(a)～(h)第一～八帧

Fig. 7 Tracking results of class Car. (a)-(h) 1st-8th frames

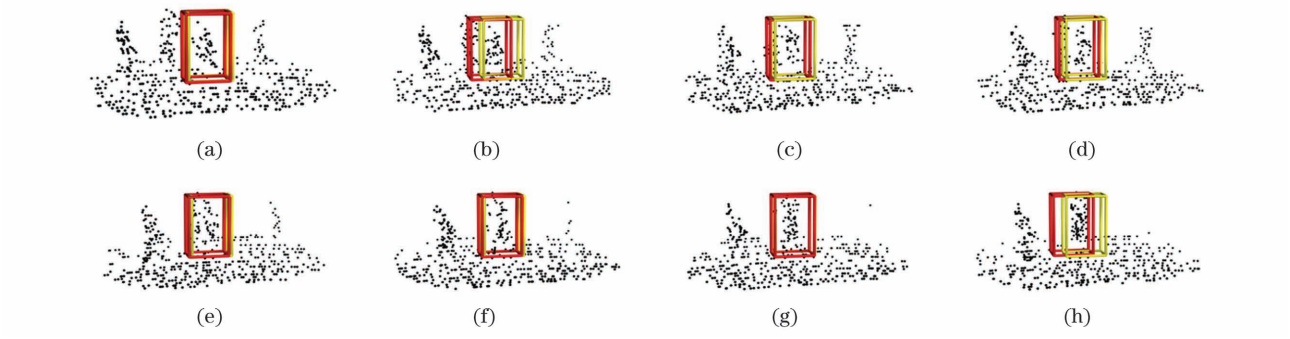

图 8 Pedestrian 类目标部分跟踪结果。(a)～(h)第一～八帧

Fig. 8 Tracking results of class Pedestrian. (a)－(h) 1st-8th frames

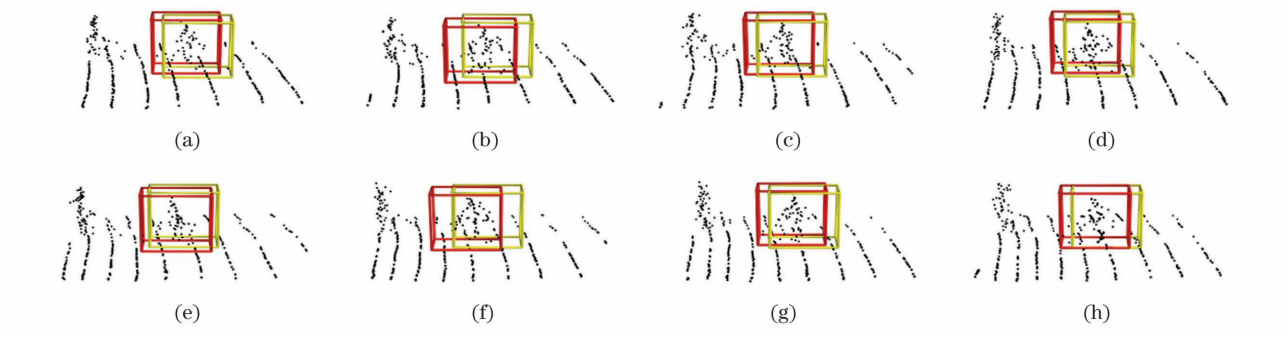

图 9 Cyclist 类目标部分跟踪结果。(a)～(h)第一～八帧

Fig. 9 Tracking results of class Cyclist. (a)－(h) 1st-8th frames

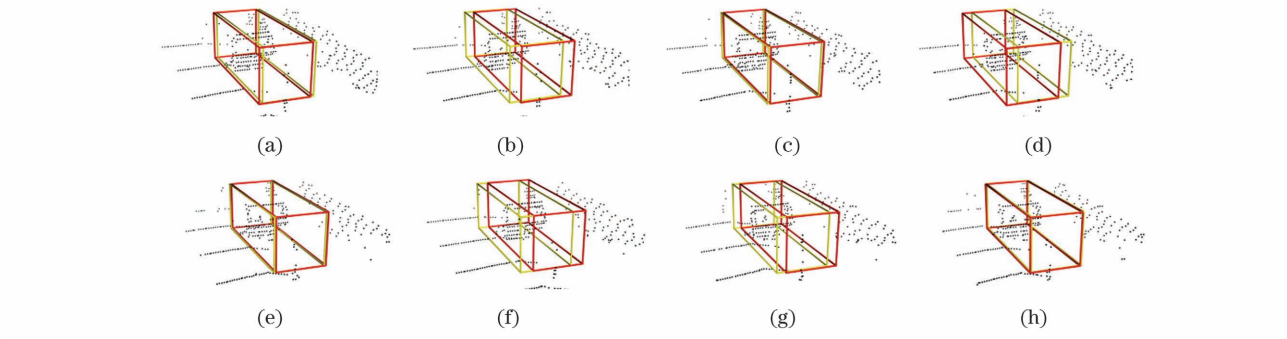

图 10 Van 类目标部分跟踪结果。(a)～(h)第一～八帧

Fig. 10 Tracking results of class Van. (a)－(h) 1st-8th frames

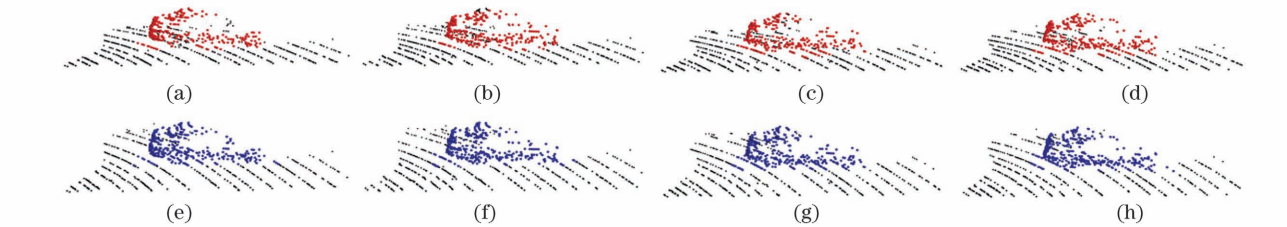

图 11 Car 类目标部分附加分割任务结果。(a)(e)第一帧;(b)(f)第二帧;(c)(g)第三帧;(d)(h)第四帧

Fig. 11 Attached segmentation task results of class Car. (a)(e) 1st frame; (b)(f) 2nd frame; (c)(g) 3rd frame; (d)(h) 4th frame

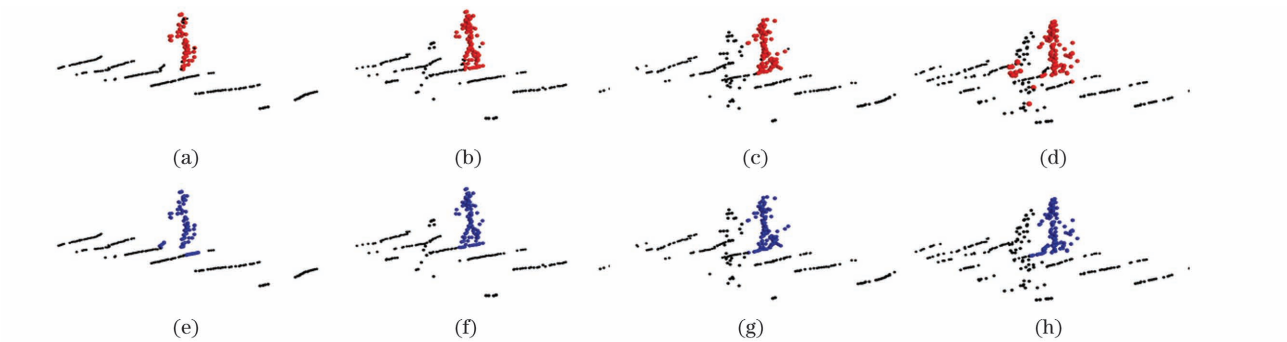

图 12 Pedestrian 类目标部分附加分割任务结果。(a)(e)第一帧;(b)(f)第二帧;(c)(g)第三帧;(d)(h)第四帧

Fig. 12 Attached segmentation task results of class Pedestrian. (a)(e) 1st frame; (b)(f) 2nd frame; (c)(g) 3rd frame; (d)(h) 4th frame

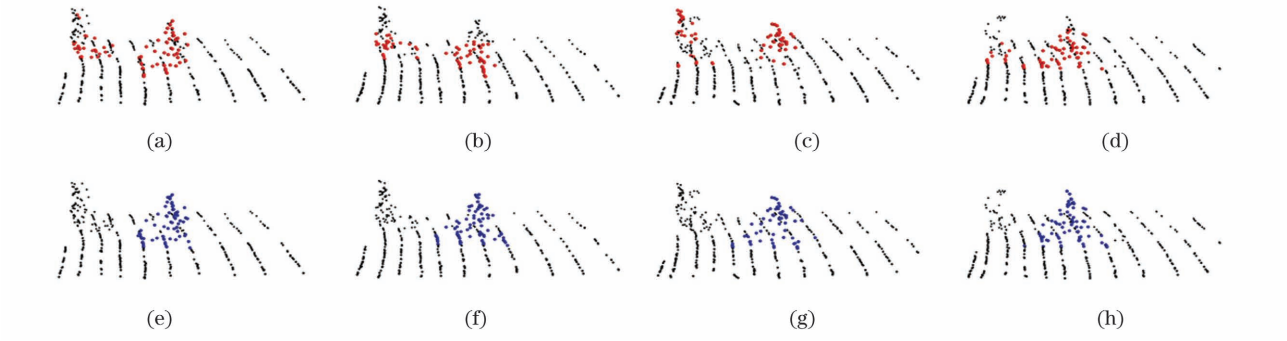

图 13 Cyclist 类目标部分附加分割任务结果。(a)(e)第一帧;(b)(f)第二帧;(c)(g)第三帧;(d)(h)第四帧

Fig. 13 Attached segmentation task results of class Cyclist. (a)(e) 1st frame; (b)(f) 2nd frame; (c)(g) 3rd frame; (d)(h) 4th frame

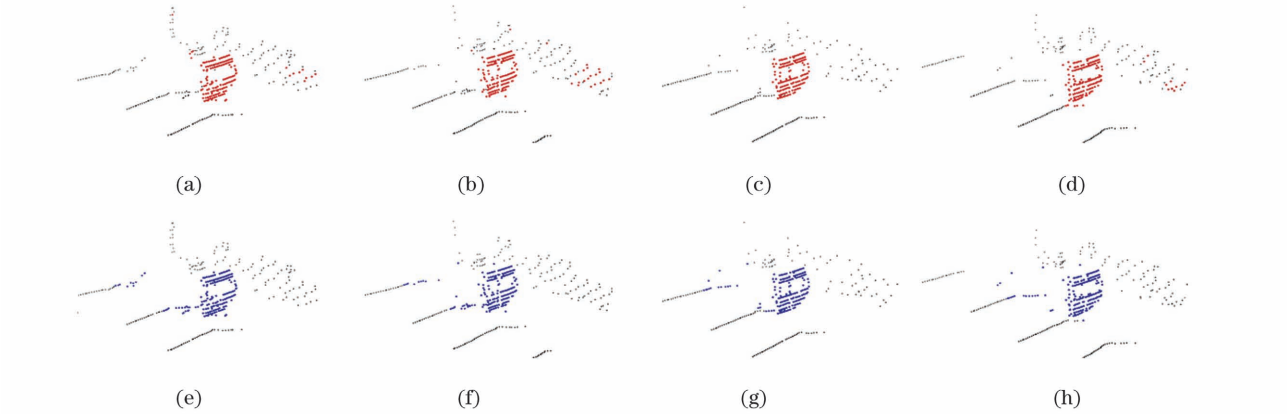

图 14 Van 类目标部分附加分割任务结果。(a)(e)第一帧;(b)(f)第二帧;(c)(g)第三帧;(d)(h)第四帧

Fig. 14 Attached segmentation task results of class Van. (a)(e) 1st frame; (b)(f) 2nd frame; (c)(g) 3rd frame; (d)(h) 4th frame

如同 P2B 算法在 Cyclist 类数据上出现的性能退化的情况，所提算法在相同设定下时，也在此类数据上表现不佳，主要原因有两方面：1)该类数据总标注数量较小，导致训练集、验证集和测试集数量均相对其他类数据更少，对于网络而言更容易出现在训练集上过拟合，而在测试集上泛化较差的情况，需要针对该数据集调整训练参数设定，如调整损失函数参数、训练轮数、学习率等，或针对性优化网络，如简化网络层数、加入 Dropout 层等；2)该类目标点云的点数均较少，使得目标模板点云和搜索区域点云能提供的目标有效信息非常有限。而所提算法参照 SC3D 和 P2B 算法的设定，对各类目标统一设定目标模板点云输入点数为 512，搜索区域点云输入点数为 1024，且采取点复制的方法进行数据增广，使得目标模板点云和搜索区域点云实际上是对少量有效点的重复。这可能会在后续处理过程中造成少量有效信息的过度学习和拟合，同时在较大点数的设定下必然造成计算资源的浪费，影响算法对小规模点云目标的处理速度。

所提算法对不同搜索区域点云数量设定下的跟踪性能进行了对比，如表 2 所示。从表中可以看出：当搜索区域点云点数量设定为512、目标模板点云

表 2 不同搜索区域点云数量 S、目标模板点云数量 T 设定下的性能对比

Table 2 Performance comparison under different number of search area point clouds S and template point clouds T unit: %

Parameter	S	T	Car 6424	Pedestrian 6088	Cyclist 308	Van 1248	Mean 14068
R_S	512	256	56.63	30.48	32.13	37.89	43.11
	512	512	57.55	36.87	32.82	44.92	46.94
	1024	512	56.98	28.40	30.44	48.60	43.29
R_P	512	256	72.60	54.14	42.62	46.31	61.62
	512	512	73.22	66.11	44.53	54.22	67.83
	1024	512	73.73	52.40	41.67	59.25	62.51

点数量设定为 512 时，所提算法对 Pedestrian 和 Cyclist 类目标的跟踪性能均得到了明显提升；当搜索区域点云点数量设定为 512、目标模板点云点数量设定为 256 时，所提算法对 Pedestrian 和 Cyclist 类目标的跟踪性能相比原设定仍有提升，但相比同时设定为 512 点时有所退化；而所提算法对 Car 和 Van 类目标的跟踪性能在两种新设定的情况下均有所降低。这说明降低搜索区域点云点数和目标模板点云点数，对于稀疏、小规模点云目标，如 Pedestrian 和 Cyclist 类，所提算法均能提升性能，因为能减少对有效信息的过度重复，避免网络对某些信息过拟合；而对于密集、较大规模点云目标，如 Car 和 Van 类，降低点数的设定则会导致所提算法跟踪性能退化，是因为有效信息随着点数减少而丢失，未能得到充分利用。

综上所述，不同目标的最佳搜索区域点云、目标模板点云数量设定是不同的，并且需要在搜索区域点云、目标模板点云数量的设定之间取得平衡，以达到所提算法对该类目标的最佳跟踪性能。

所提算法在 P2B 算法中采用的附加网络设计增大了网络的参数量，如表 3 所示。综上，原 P2B 算法网络参数量为 2.8 MB，所提算法附加网络部分参数量为 0.3 MB，总网络参数量为 3.1 MB。由此可见，算法增加的网络参数量占总网络参数量的比例较小，以较小的网络模型复杂度代价换来了明显的性能提升。

表 3 附加网络各网络层参数量

Table 3 Number of parameters of each network layer of auxiliary network

Module	Layer name	Number of input channels	Number of output channels	Kernel size	Stride size	Parameter size
Aux_module	3-NN	256	256			0
	Interpolation	128	256			0
	Conv2d	512	256	(1, 1)	(1, 1)	131072
	BatchNorm2d	256	256			512
	ReLU	256	256			0
	3-NN	512	512			0
	Interpolation	256	512			0
	Conv2d	384	256	(1, 1)	(1, 1)	98304
	BatchNorm2d	256	256			512
	ReLU	256	256			0
	3-NN	1024	1024			0
	Interpolation	512	1024			0
	Conv2d	256	256	(1, 1)	(1, 1)	65536
	BatchNorm2d	256	256			512
	ReLU	256	256			0
Aux_regreesion	Linear	256	3			768
Aux_segmentation	Linear	256	1			256
Total						297472

4.3 算法实时性评估

在利用 KITTI 数据集对各算法的评估指标进行测试对比的同时，对各算法的平均速度进行了记录，如表 4 所示。由于采用了与 P2B 算法不同的实验平台，原 P2B 算法的速度由原来的 45.5 frame·s^{-1}[9] 提升到了 55.6 frame·s^{-1}。由表可知，所提算法由于在网络推断阶段的网络规模实际与 P2B 算法网络一致，只是在训练阶段接入附加网络辅助引导骨干网络，更好地学习了稀疏小规模点云目标的特征，因此在推断时算法平均速度与 P2B 算法相比没有明显损失。

表 4 各算法平均速度对比

Table 4 Average speed comparison of various methods

Method	SC3D	P2B	Proposed algorithm
Speed /(frame·s^{-1})	2.2	55.6	55.6

5 结 论

提出融合附加神经网络的激光雷达点云单目标跟踪算法。针对初始帧目标模板点云稀疏、规模小、有效信息少的情况，设计了附加神经网络，执行附加的逐点前景背景分割任务和与目标中心点坐标偏移

量的回归任务，增强了对此类目标细粒化、内在结构化特征的学习能力。同时，所设计的附加网络仅在网络训练阶段对原有骨干网络进行优化监督，在推断阶段不参与数据处理，使得所提算法在原有算法的基础上以增加少量网络参数量的代价，取得了跟踪性能的提升和实时性的保持。通过实验验证可得，相较于原 P2B 算法，所提算法在相同的 KITTI 数据集上平均跟踪成功率提高了 0.89 个百分点，平均跟踪准确率提高了 2.51 个百分点，尤其在 Pedestrian 类数据上平均跟踪准确率提高了 2.80 个百分点。针对依照原有算法设定搜索区域点云数量、目标模板点云数量时，所提算法对 Cyclist 类数据跟踪性能有所退化的情况，讨论了不同的搜索区域点云数量、目标模板点云数量设定对各类目标跟踪性能的影响，最终通过合理的调整设定方式，所提算法对 Cyclist 类数据跟踪成功率提高了 2.38 个百分点，跟踪准确率提高了 2.86 个百分点；对 Pedestrian 类数据跟踪成功率提高了 8.47 个百分点，跟踪准确率提高了 13.71 个百分点，在这两类稀疏、小规模点云目标的跟踪性能上相比原设定进步明显。所提算法虽然增加了少量的网络模型参数量，但算法平均速度仍然保持在 55.6 frame$\cdot s^{-1}$，达到了预期的效果。

综上所述，所提算法在预期解决的稀疏小规模点云单目标跟踪问题上表现良好，取得了明显的性能提升，同时附加网络的思想可扩展至更多同类算法和任务中，具有一定的研究价值和空间。分析实验所得的可视化结果发现，搜索区域点云中的同类干扰物对跟踪结果有影响，导致预测框对目标定位不准确。如何利用搜索区域点云背景信息和有效更新搜索区域点云是进一步的研究方向。

参 考 文 献

[1] Lei X D, Wang H T, Zhao Z Z. Small sample airborne LiDAR point cloud classification based on transfer learning[J]. Chinese Journal of Lasers, 2020, 47(11): 1110002.
雷相达, 王宏涛, 赵宗泽. 基于迁移学习的小样本机载激光雷达点云分类[J]. 中国激光, 2020, 47(11): 1110002.

[2] Gu S T, Wang L, Ma Y X, et al. Local feature description of LiDAR point cloud data based on hierarchical Mercator projection[J]. Acta Optica Sinica, 2020, 40(20): 2015001.
顾尚泰, 王玲, 马燕新, 等. 基于分层墨卡托投影的激光雷达点云数据局部特征描述[J]. 光学学报, 2020, 40(20): 2015001.

[3] Hu H Y, Hui Z Y, Li N. Airborne LiDAR point cloud classification based on multiple-entity eigenvector fusion[J]. Chinese Journal of Lasers, 2020, 47(8): 0810002.
胡海瑛, 惠振阳, 李娜. 基于多基元特征向量融合的机载 LiDAR 点云分类[J]. 中国激光, 2020, 47(8): 0810002.

[4] Yi Q, Zhong H Y, Liu L, et al. Dynamic inspection of wheel profile based on ROI-RSICP algorithm[J]. Chinese Journal of Lasers, 2020, 47(11): 1104006.
易倩, 钟浩宇, 刘龙, 等. 基于 ROI-RSICP 算法的车轮廓形动态检测[J]. 中国激光, 2020, 47(11): 1104006.

[5] Fan X H, Xu G L, Li W L, et al. Target segmentation method for three-dimensional LiDAR point cloud based on depth image[J]. Chinese Journal of Lasers, 2019, 46(7): 0710002.
范小辉, 许国良, 李万林, 等. 基于深度图的三维激光雷达点云目标分割方法[J]. 中国激光, 2019, 46(7): 0710002.

[6] Zhang Z J, Cheng X J, Cao Y J, et al. Application of 3D reconstruction of relic sites combined with laser and vision point cloud[J]. Chinese Journal of Lasers, 2020, 47(11): 1110001.
张子健, 程效军, 曹宇杰, 等. 结合激光与视觉点云的古遗迹三维重建应用[J]. 中国激光, 2020, 47(11): 1110001.

[7] Zarzar J, Giancola S, Ghanem B. Efficient bird eye view proposals for 3D Siamese tracking[EB/OL]. (2019-03-25)[2021-02-22]. https://arxiv.org/abs/1903.10168.

[8] Giancola S, Zarzar J, Ghanem B. Leveraging shape completion for 3D Siamese tracking[C]//2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 15-20, 2019, Long Beach, CA, USA. New York: IEEE Press, 2019: 1359-1368.

[9] Qi H Z, Feng C, Cao Z G, et al. P2B: point-to-box network for 3D object tracking in point clouds[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 13-19, 2020, Seattle, WA, USA. New York: IEEE Press, 2020: 6328-6337.

[10] Wang N Y, Shi J P, Yeung D Y, et al. Understanding and diagnosing visual tracking systems[C]//2015 IEEE International Conference on Computer Vision (ICCV), December 7-13, 2015, Santiago, Chile. New York: IEEE Press, 2015: 3101-3109.

[11] Achlioptas P, Diamanti O, Mitliagkas I, et al. Learning representations and generative models for 3d

point clouds[EB/OL]. (2017-07-08)[2021-02-20]. https://arxiv.org/abs/1707.02392.

[12] Li B, Yan J J, Wu W, et al. High performance visual tracking with Siamese region proposal network[C]// 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 18-23, 2018, Salt Lake City, UT, USA. New York: IEEE Press, 2018: 8971-8980.

[13] Zou H, Cui J H, Kong X, et al. F-Siamese tracker: a frustum-based double Siamese network for 3D single object tracking[C]// 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 24 -January 24, 2021, Las Vegas, NV, USA. New York: IEEE Press, 2021: 8133-8139.

[14] Fang Z, Zhou S F, Cui Y B, et al. 3D-SiamRPN: an end-to-end learning method for real-time 3D single object tracking using raw point cloud[J]. IEEE Sensors Journal, 2021, 21(4): 4995-5011.

[15] Qi C R, Litany O, He K M, et al. Deep Hough voting for 3D object detection in point clouds[C]// 2019 IEEE/CVF International Conference on Computer Vision (ICCV), October 27-November 2, 2019, Seoul, Korea (South). New York: IEEE Press, 2019: 9276-9285.

[16] Qi C R, Yi L, Su H, et al. PointNet++: deep hierarchical feature learning on point sets in a metric space[C]// NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems, December 4-9, Long Beach, CA, USA. Red Hook: Curran Associates, 2017: 5105-5114.

[17] He C H, Zeng H, Huang J Q, et al. Structure aware single-stage 3D object detection from point cloud[C]// 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 13-19, 2020, Seattle, WA, USA. New York: IEEE Press, 2020: 11870-11879.

[18] Geiger A, Lenz P, Urtasun R. Are we ready for autonomous driving? The KITTI vision benchmark suite[C]//2012 IEEE Conference on Computer Vision and Pattern Recognition, June 16-21, 2012, Providence, RI, USA. New York: IEEE Press, 2012: 3354-3361.

[19] Girshick R. Fast R-CNN[C]// 2015 IEEE International Conference on Computer Vision (ICCV), December 7-13, 2015, Santiago, Chile. New York: IEEE Press, 2015: 1440-1448.

[20] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C]//2017 IEEE International Conference on Computer Vision (ICCV), October 22-29, 2017, Venice, Italy. New York: IEEE Press, 2017: 2999-3007.

[21] Kristan M, Matas J, Leonardis A, et al. A novel performance evaluation methodology for single-target trackers[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016, 38(11): 2137-2155.

Single Object Tracking of LiDAR Point Cloud Combined with Auxiliary Deep Neural Network

Zhou Xiaoyu[1], Wang Ling[1*], Ma Yanxin[2], Chen Peibo[1]

[1] *College of Electronic Science, National University of Defense Technology, Changsha, Hunan* 410073, *China;*

[2] *College of Meteorology and Oceanography, National University of Defense Technology, Changsha, Hunan* 410073, *China*

Abstract

Objective Point cloud can be selected as an ideal data format for tasks such as object classification, detection, segmentation, reconstruction, and tracking in a three-dimensional (3D) scene. In the case of a single object tracking task, the approach of considering point clouds as data outperforms that of selecting two-dimensional (2D) picture or video sequences for two reasons. First, the point cloud can better describe the 3D geometric information of the object in real scenes, such as the position, scale, and posture of the object. Second, different from the passive optical imaging principle of cameras, information is collected using light detection and ranging (LiDAR) following an active imaging approach, which is not prone to be affected by natural light conditions. Therefore, the point cloud can adapt to different conditions involving visual degradation or illumination and is robust to glare, reflections, and shadows. Based on this discussion, the single object tracking of 3D point cloud is a topic worth investigating. Generally, single object tracking tasks aim to use the information in the given initial frame to determine the tracked

object and predict the locating bounding box of the object in each subsequent frame. However, existing single object trackers of LiDAR point clouds exhibit a poor tracking performance of sparsely distributed and small-scale point cloud objects. This is mainly attributed to the downscaling operation applied to features extracted from the point cloud, leading to the insufficient application of object's structural information; this distracts the tracker from performing accurate bounding box predictions of sparsely distributed and small-scale point cloud objects.

Methods To address this problem, a single object tracking network combined with auxiliary deep neural network is proposed herein. During the training stage, we attach a modified auxiliary network to the backbone network, which accomplishes two auxiliary tasks: 1) foreground point cloud segmentation, which guides the backbone network to focus on pointwise semantic information; 2) pointwise center coordinate offset regression, which leads the features to be aware of the intrastructural information of the object. These two tasks are jointly supervised using the backbone network such that the semantic and structural features are naturally stored in the object features extracted using the backbone network. However, during the inference stage, the auxiliary network is bypassed in this process because the trained backbone network is already optimized to be structure aware and detaching the auxiliary network can avoid extra computational cost, which is essential for retaining the real-time performance of the tracker. Moreover, we notice that the latest work follows the same manner as the dataset organization. In particular, the number of input points in the search area point cloud and template point cloud is fixed, irrespective of the class of point cloud data. However, as the KITTI dataset presents, the point cloud of some classes is dense and comprises a large number of points, while the point cloud of other classes suffers from scarce points, providing insufficient and limited object information. A fixed number of input points may be unsuitable for all data classes. Hence, we propose setting different input quantities for each class during both the training and inference stages, which is accomplished without changing the network structure. In general, the structure of different network modules is shown in Fig. 1 to Fig. 6 respectively.

Results and Discussions Both qualitative and quantitative experiments are conducted to prove the superiority of our propositions. Table 1 shows the result of extensive comparisons between our proposed tracker and other two former trackers; our tracker achieves better results in three of four data classes and shows higher mean performance than the other two former trackers. Test results on the KITTI dataset show that our network increases the average tracking success by 0.89 percent and the average tracking accuracy by 2.51 percent under the same parameter settings as the existing work. Second, some specific tracking results of four data classes are depicted in Fig. 7–Fig. 10. These figures show that our tracker can predict bounding boxes closely similar to the ground truth. Furthermore, we present some results of the tasks processed using the auxiliary network. Concretely, Fig. 11–Fig. 14 show the results of foreground segmentation task, in which our auxiliary network can accurately segment the surface of the object from the background points. Moreover, details of comparisons on tracking performance with different numbers of the search area point cloud and template point cloud are discussed in Table 2, which prove the efficiency of adjusting a suitable quantity of input points of different data classes. Compared with proposed algorithm, the average tracking success increases by 4.54 percent, and the average tracking accuracy increases by 7.83 percent. In particular, for class Cyclist, the average tracking success increases by 2.38 percent and the average tracking accuracy increases by 2.86 percent; for class Pedestrian, the average tracking success increases by 8.47 percent and the average tracking accuracy increases by 13.71 percent. These findings imply that our tracker achieves improvements in terms of the tracking performance of sparse and small-scale objects. Finally, Table 3 and Table 4 show that we succeed in maintaining a balance between performance and computational costs.

Conclusions In summary, the proposed method achieves reasonable results in addressing the problem of tracking sparsely distributed and small-scale point cloud objects as expected, and can be applied to solve other tasks. Inspired by our experiment results, to achieve further improvement, we will seek new approaches on data augmentation and extract more useful clues using the background information for search area updates.

Key words image processing; LiDAR; point cloud; auxiliary network; single object tracking

OCIS codes 100.6890; 280.3640; 150.6910; 150.0155