西安電子科技大學

基于点云语义分割的货车装载量检测算法研究

薛佳瑶

目录

01 研究背景及意义

02 货车点云数据采集

03 基于点云数据处理的货车分割

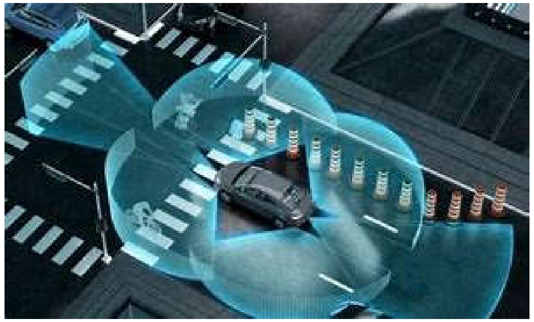
04 曲面重建及体积计算

05 总结与展望

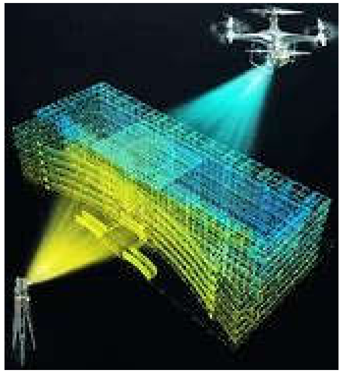
研究背景及现状

研究背景及意义

智能家居



自动驾驶



测绘

激光雷达优点：测量精度高、响应速度快、抗干扰性强

使用三维激光扫描技术捕获处理建筑工地、建筑工程和建筑设备的3D点云数据，获得准确有效代表三维表面几何形状的建筑应用数据

研究背景及意义

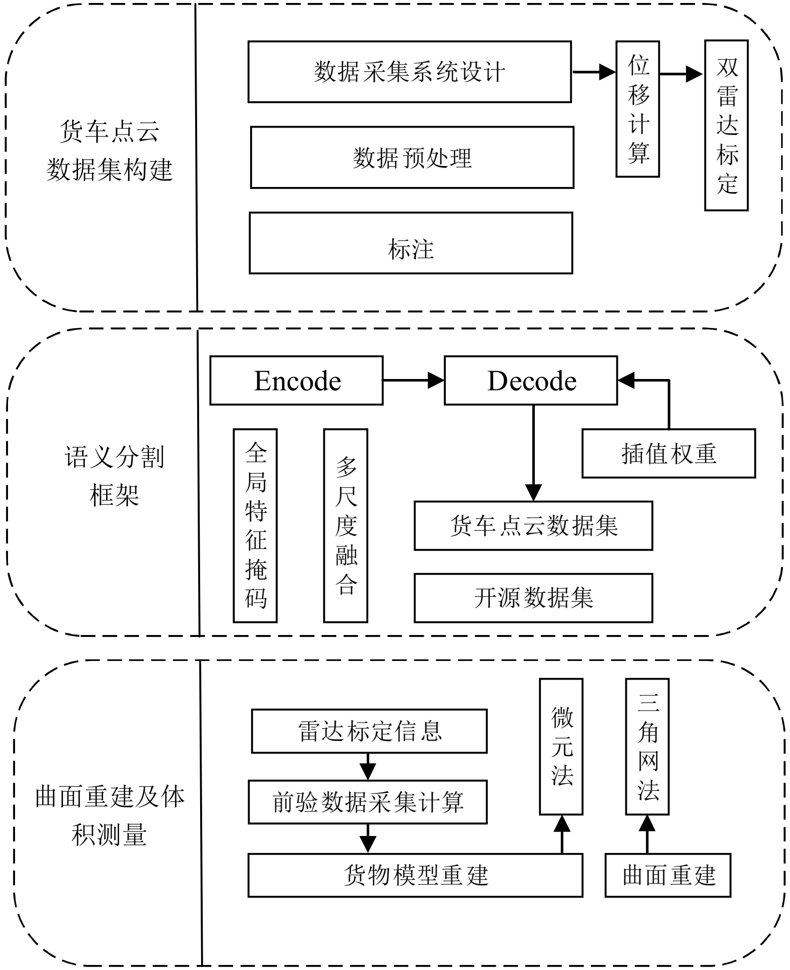
传统的语义分割方法

- 边缘
- 区域分割
- 属性
- 模型

基于深度学习的自动语义分割方法

- 基于视图和投影
- 基于体素化
- 基于点处理

论文主要技术路线


货车点云
数据集构建

数据采集系统设计

位移计算

双雷达标定

数据预处理

标注

语义分割
框架

Encode

Decode

插值权重

全局特征掩码

多尺度融合

货车点云数据集

开源数据集

曲面重建及体
积测量

雷达标定信息

前验数据采集计算

货物模型重建

微元法

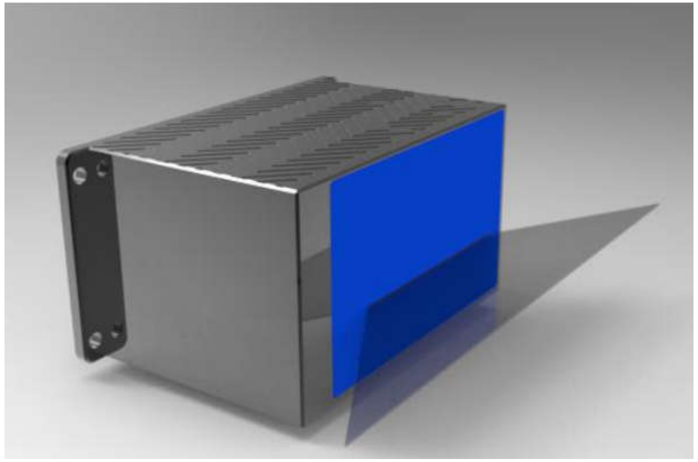
三角网法

曲面重建

货车点云数据采集

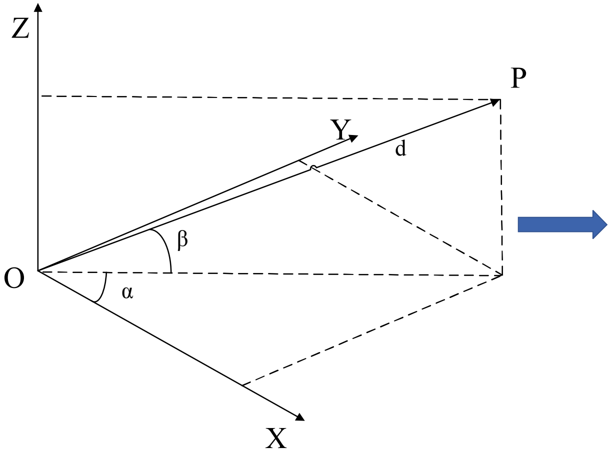
设计货车点云数据采集系统，对工地过地磅处的行驶货车进行数据采集以及数据预处理，制作完整的货车数据集便于后续点云分割任务。

货车点云数据采集



单线激光雷达HS1	
波长	905nm
测量范围	100m（反射率10%）
测量精度	±2cm
扫描频率	80~200Hz
视场角	120°
水平角分辨率	80Hz：0.18° 200Hz：0.44°

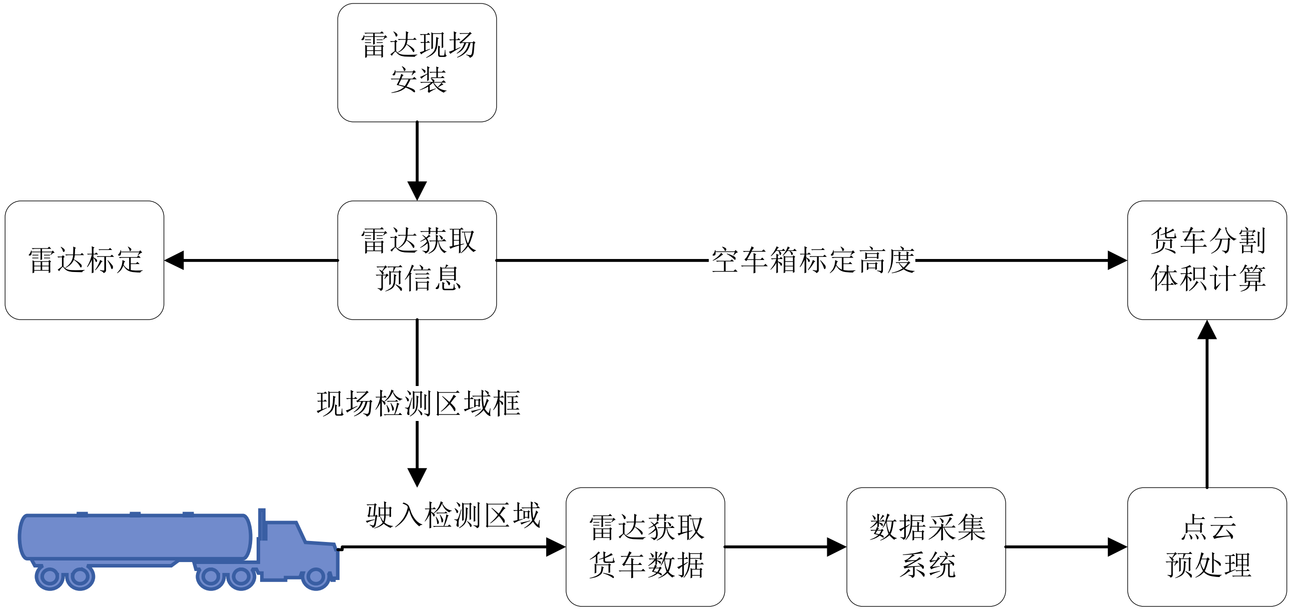
飞行时间测量法： $distance = c\times(t_2-t_1)/2$



$$P(x,y,z)=\begin{cases} x=d\times\cos\alpha\times\cos\beta \\ y=d\times\sin\alpha\times\cos\beta \\ z=d\times\sin\beta \end{cases}$$

通过数据包解析参数，可以计算出目标在雷达座标系中的三维空间位置，这些参数包括距离（d）、横向扫描角度（α）、纵向扫描角度（β）以及表面点的反射强度信息（Intensity）

货车点云数据采集流程



雷达现场安装

雷达获取预信息

雷达标定

空车箱标定高度

货车分割体积计算

现场检测区域框

驶入检测区域

雷达获取货车数据

数据采集系统

点云预处理

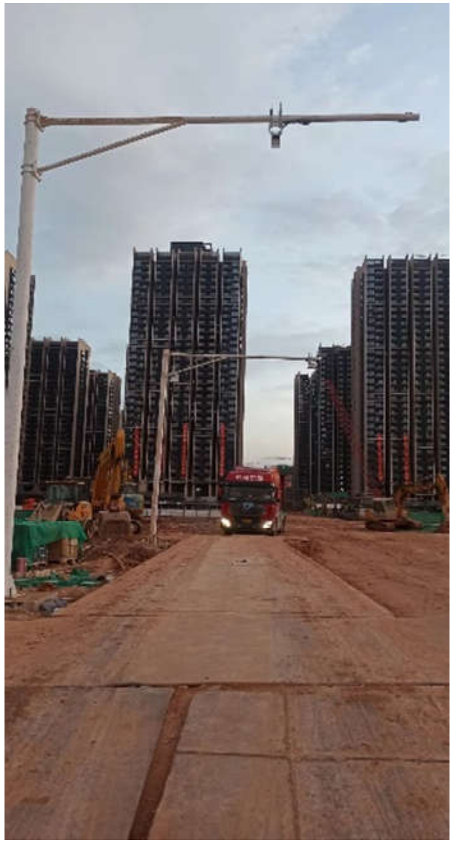
采集目标

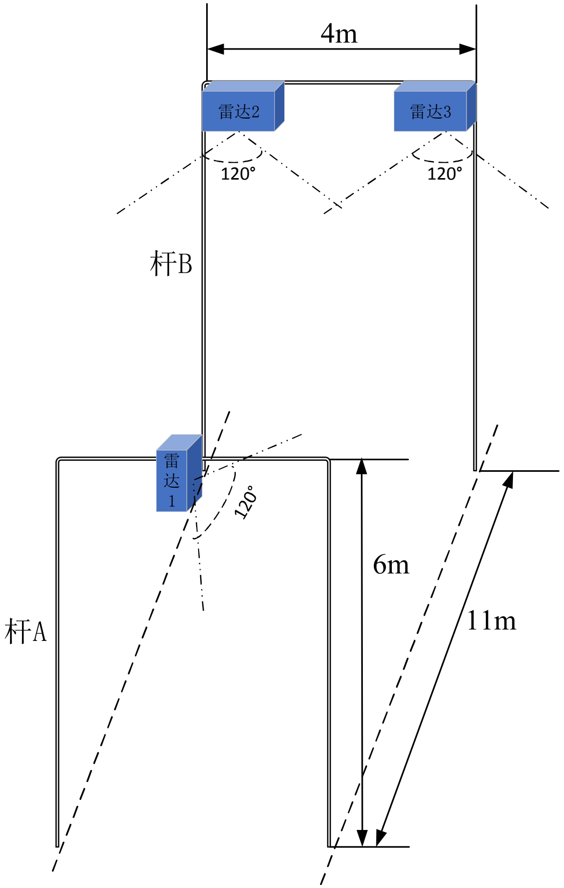
工地渣土车：东风350

长8550mm，宽2500mm，车高3450mm

货箱长度为5800mm，货箱宽度为2300mm，货箱高度为1175mm

点云数据采集方案




4m
雷达2
雷达3
120°
120°
杆B
雷达1
120°
杆A
6m
11m

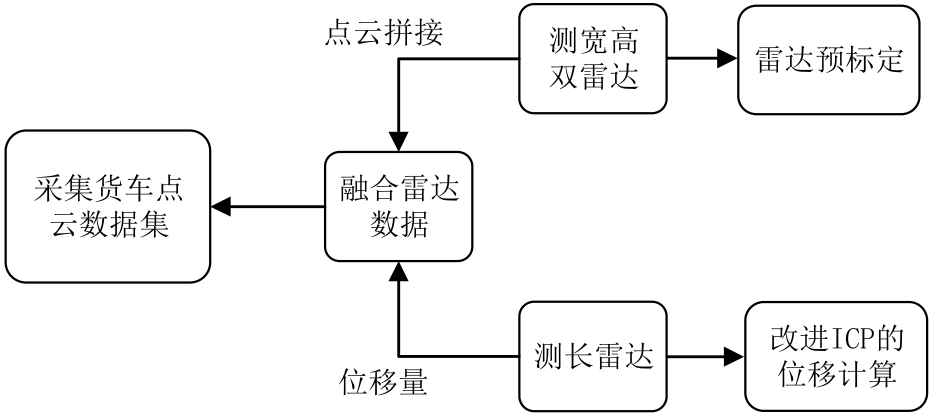
1、采集数据对象：工地渣土车

8550×2500×3450

2、数据融合

3、雷达预标定

多雷达数据融合



点云拼接

测宽高双雷达

雷达预标定

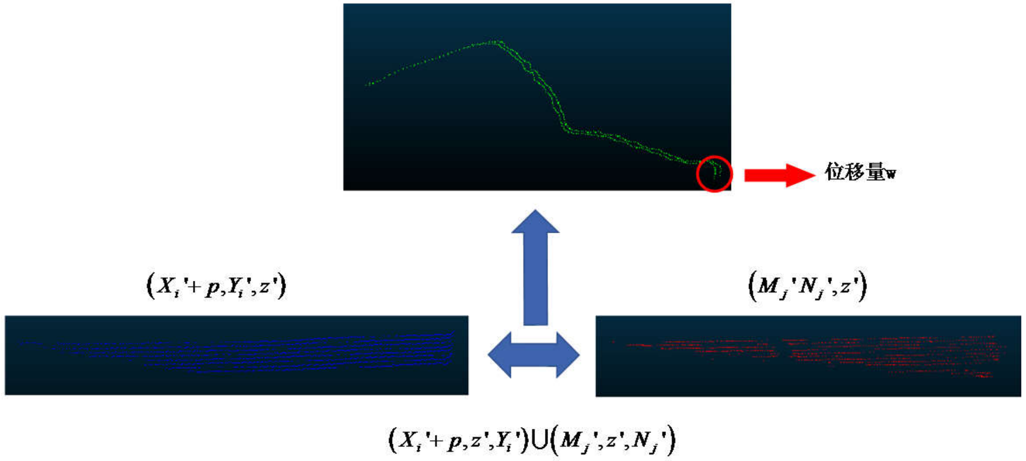
采集货车点云数据集

融合雷达数据

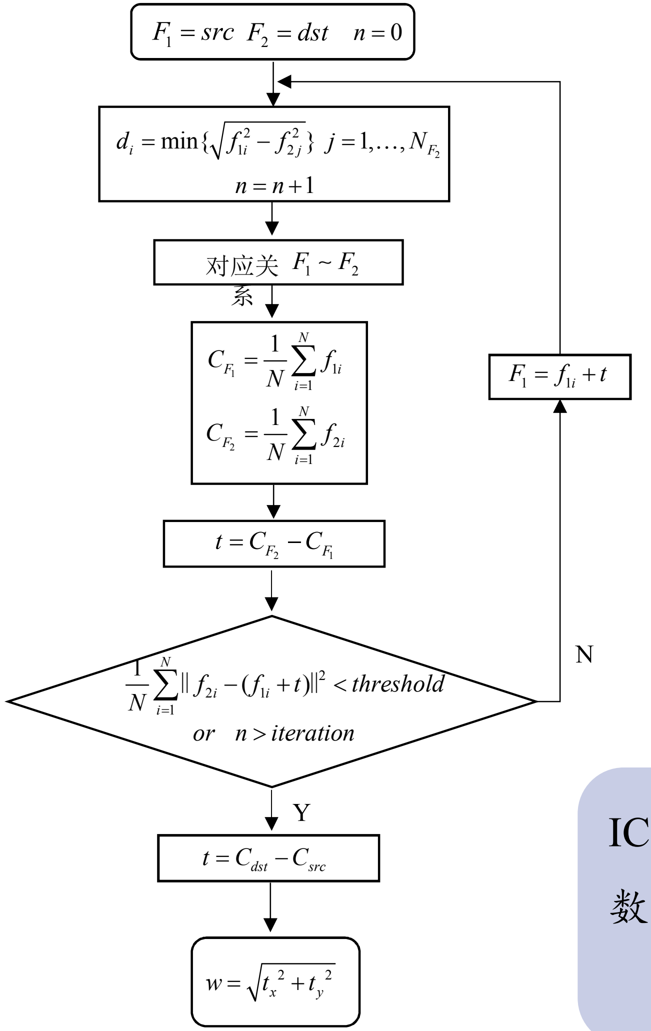
测长雷达

改进ICP的位移计算

位移量


位移量w
$(X_i'+p,Y_i',z')$
$(M_j'N_j',z')$
$(X_i'+p,z',Y_i')\cup(M_j',z',N_j')$

改进ICP的位移计算



$F_1 = src \quad F_2 = dst \quad n = 0$

$d_i = \min\{\sqrt{f_{1i}^2 - f_{2j}^2}\} \quad j = 1,\ldots,N_{F_2}$

$n = n + 1$

对应关系 $F_1 \sim F_2$

$C_{F_1} = \frac{1}{N}\sum_{i=1}^{N} f_{1i}$

$C_{F_2} = \frac{1}{N}\sum_{i=1}^{N} f_{2i}$

$t = C_{F_2} - C_{F_1}$

$\frac{1}{N}\sum_{i=1}^{N} \| f_{2i} - (f_{1i} + t)\|^2 < threshold$

$or \quad n > iteration$

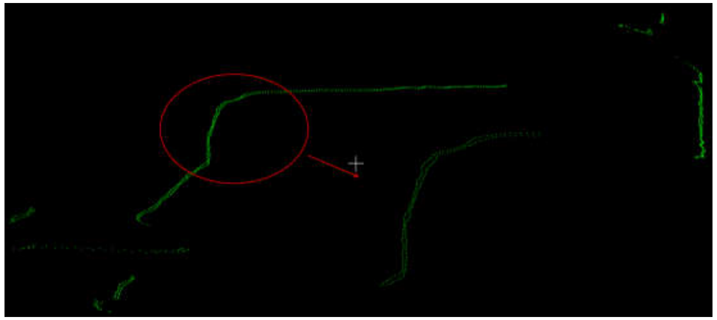
N

$F_1 = f_{1i} + t$

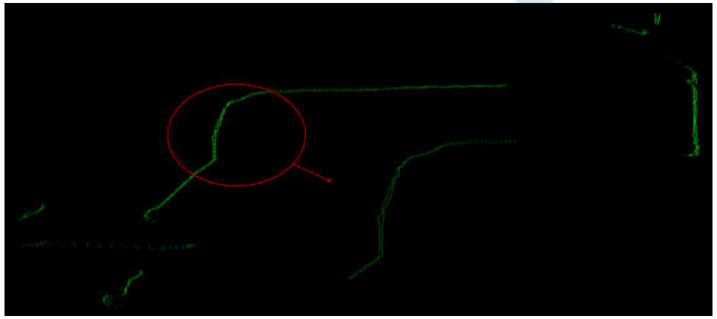
Y

$t = C_{dst} - C_{src}$

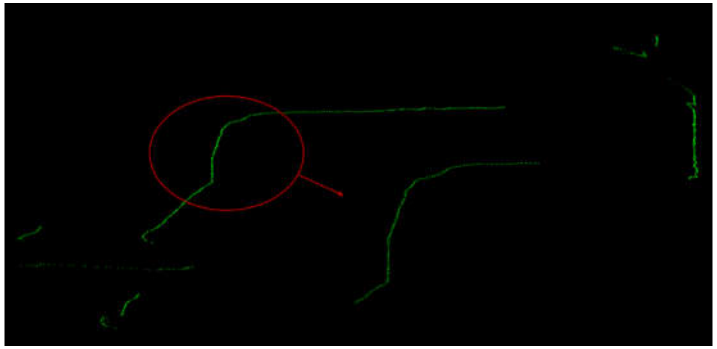
$w = \sqrt{t_x{}^2 + t_y{}^2}$



(a)测长雷达前后帧点云



(b)一次迭代



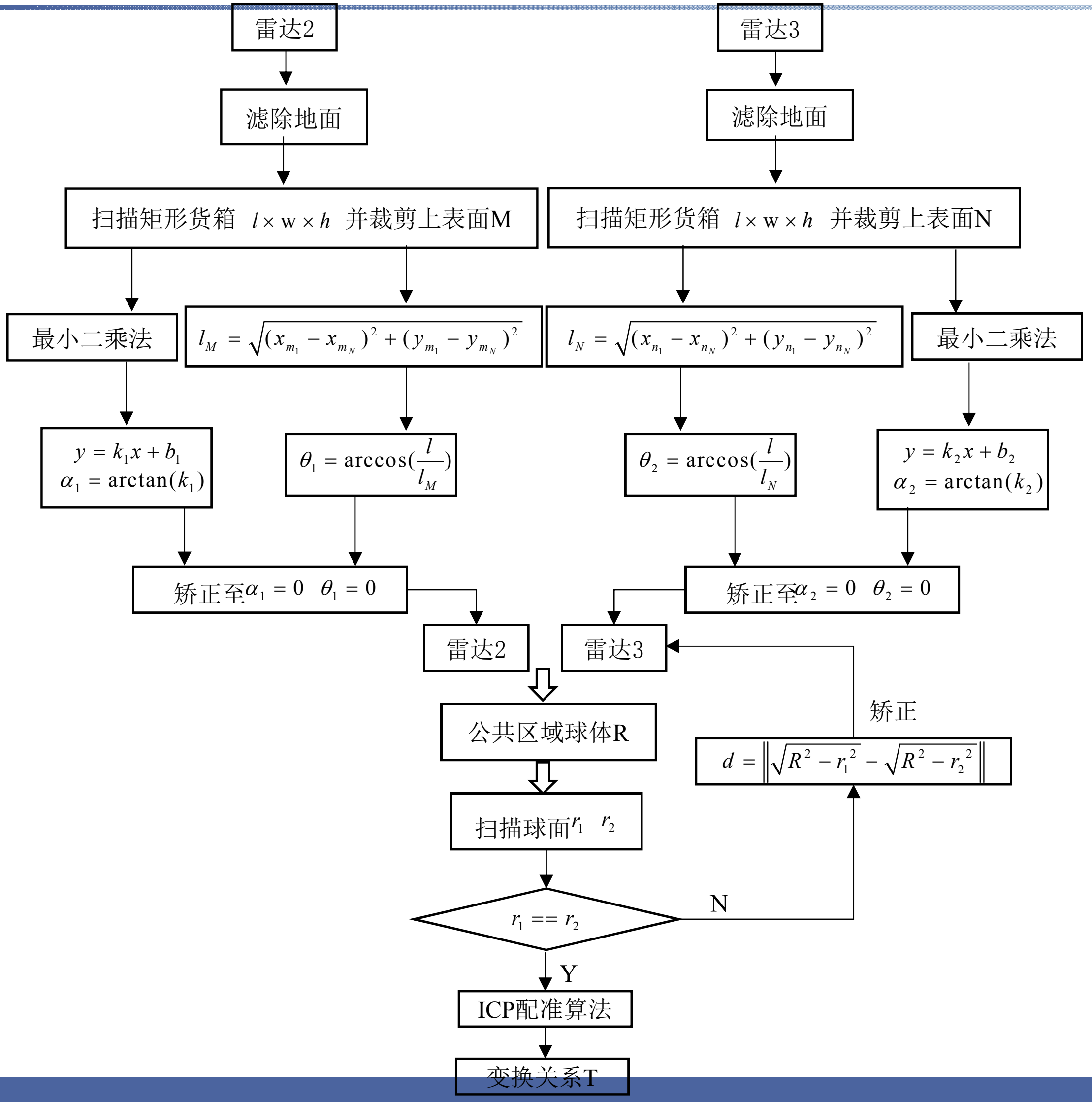
(c)11次迭代

$w = \sqrt{\Delta x^2 + \Delta y^2} = 0.293\text{mm}$

ICP（Iterative Closest Point）算法常用于三维点云数据匹配。其目标是将一个点云数据集与另一个点云数据集进行对齐，使得它们之间的误差最小化。

$$E(R,t) \propto \frac{1}{N}\sum_{i=1}^{N} \| m_i - (Rs_i + t)\|^2$$

雷达预标定



雷达2

滤除地面

扫描矩形货箱 $l\times w\times h$ 并裁剪上表面M

最小二乘法

$y=k_1x+b_1$

$\alpha_1=\arctan(k_1)$

$l_M=\sqrt{(x_{m_1}-x_{m_N})^2+(y_{m_1}-y_{m_N})^2}$

$\theta_1=\arccos(\frac{l}{l_M})$

矫正至$\alpha_1=0$ $\theta_1=0$

雷达3

滤除地面

扫描矩形货箱 $l\times w\times h$ 并裁剪上表面N

$l_N=\sqrt{(x_{n_1}-x_{n_N})^2+(y_{n_1}-y_{n_N})^2}$

$\theta_2=\arccos(\frac{l}{l_N})$

最小二乘法

$y=k_2x+b_2$

$\alpha_2=\arctan(k_2)$

矫正至$\alpha_2=0$ $\theta_2=0$

雷达2

雷达3

公共区域球体R

扫描球面r_1 r_2

$r_1==r_2$

N

$d=\left\|\sqrt{R^2-r_1^2}-\sqrt{R^2-r_2^2}\right\|$

矫正

Y

ICP配准算法

变换关系T

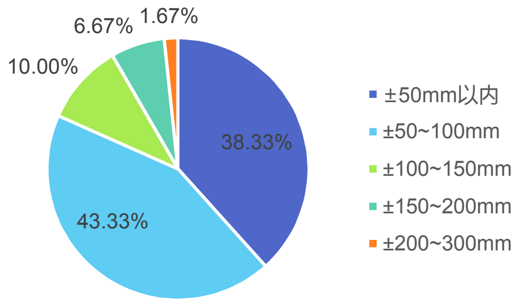
主要从三个方面进行标定

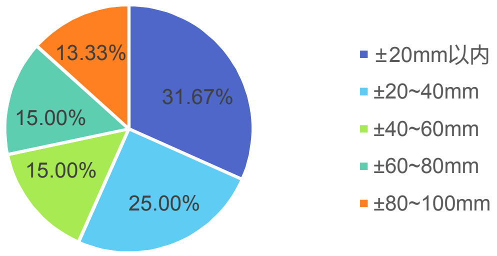
1、需确保两个雷达的扫描平面分别垂直于地磅平面，$\alpha_1=0,\alpha_2=0$

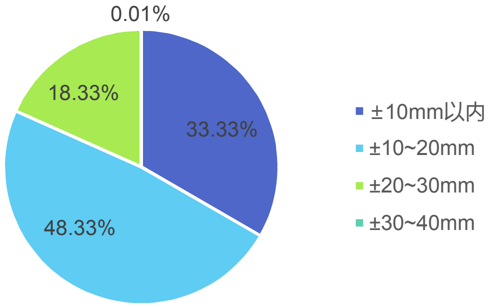
2、其次两个雷达的扫描平面分别垂直于货车行驶方向，$\theta_1=0,\theta_2=0$

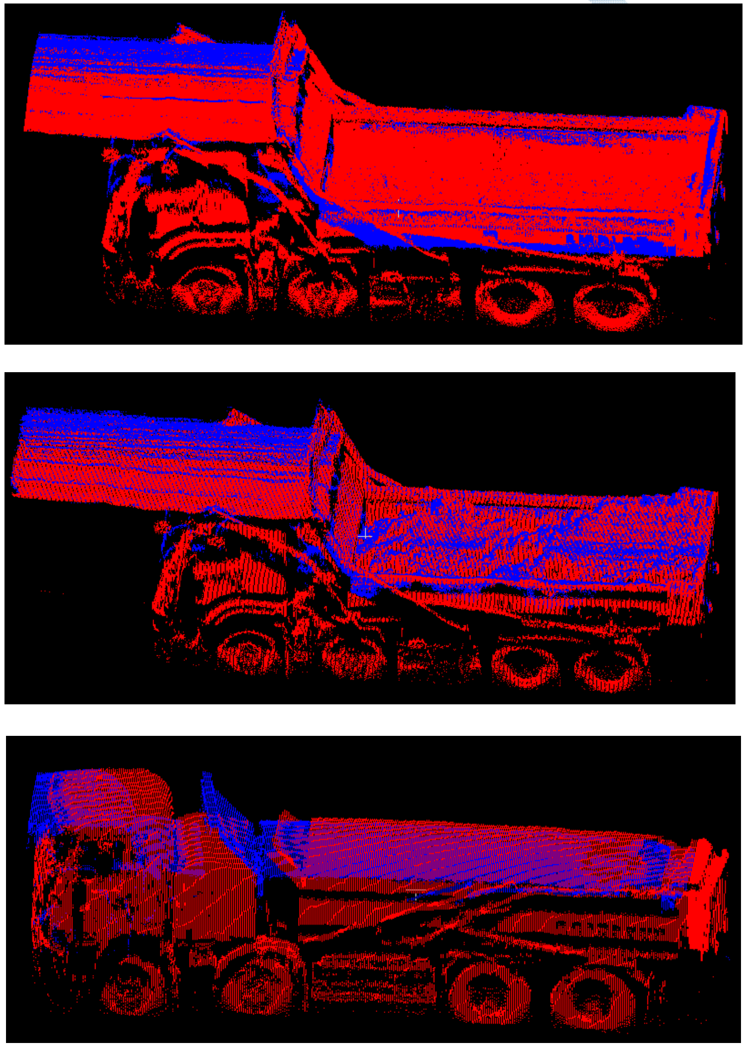
3、将测宽高双雷达通过标定置于同一水平线以及获得其之间的变换矩阵T。

货车点云数据



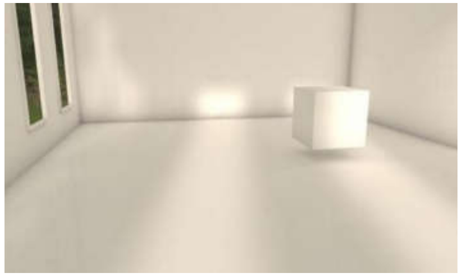






采集30组以不同速度经过过地磅处的点云数据

噪声干扰

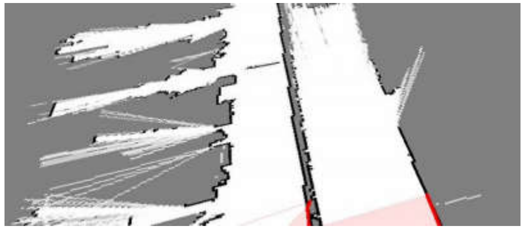


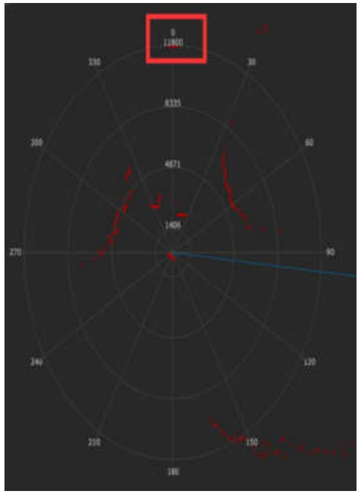
01

光源干扰，自然、人造光源产生噪点，影响传感器误触发

02

玻璃干扰，出现漏检、毛刺、扭曲和不完整的情况





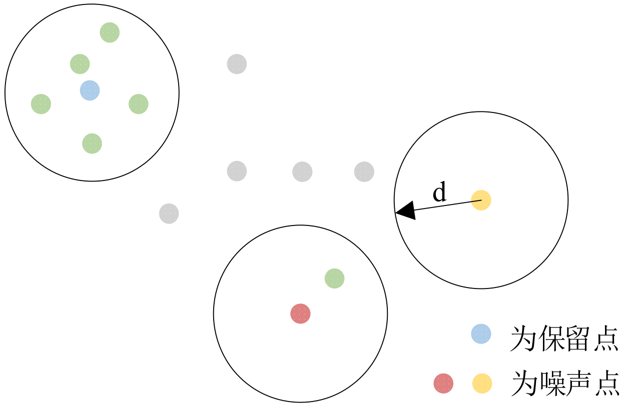
03

高、低反射物体会导致出现点云密度增加或模糊不清、漏点散点

04

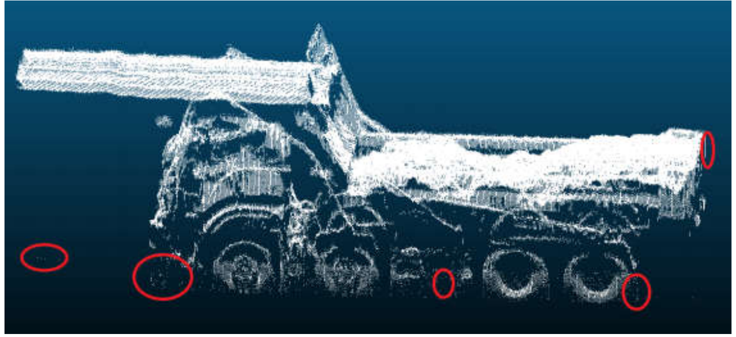
雨、雪、灰尘等对激光有遮挡效应，造成离散的孤立点和干扰点

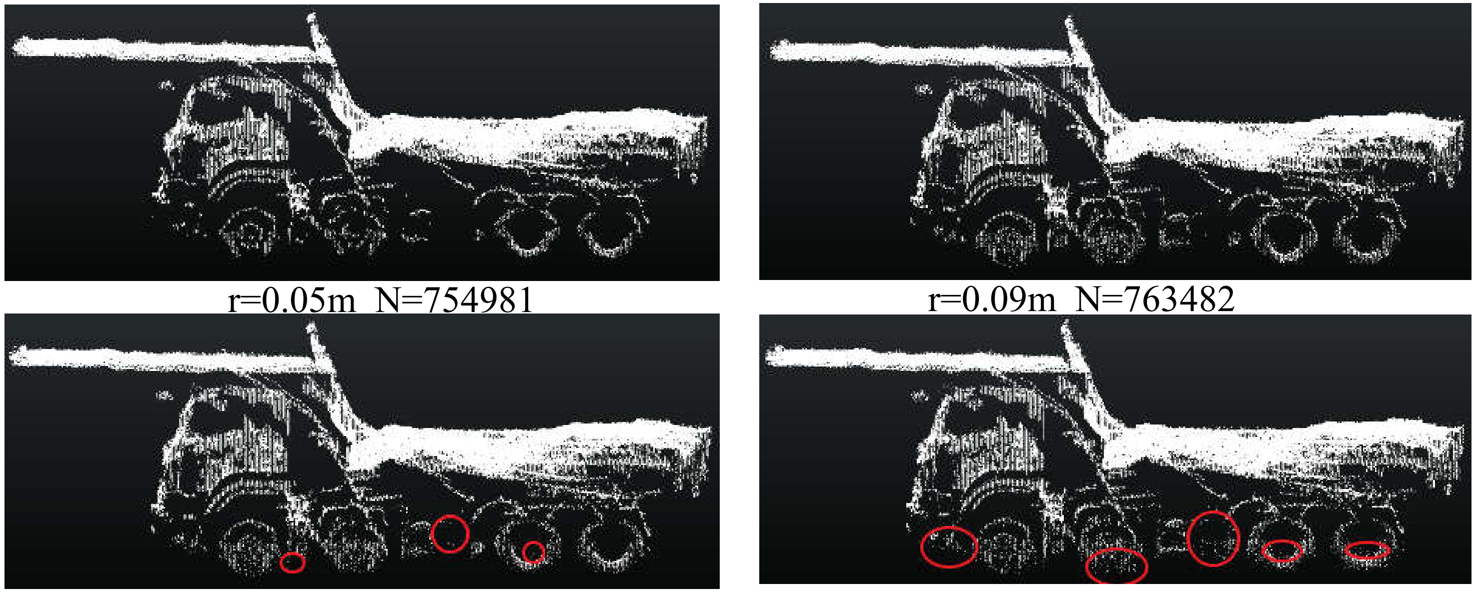
货车点云数据处理



半径滤波，是基于连通分析的点云滤波，对点云中的每一个点 P_i，确定一个半径为r的邻域，若邻域范围内的点数小于阈值，$N < N_{threshold}$，则认为点 P_i 为噪声点，并剔除

$N_{threshold} = 15$





r=0.05m N=754981

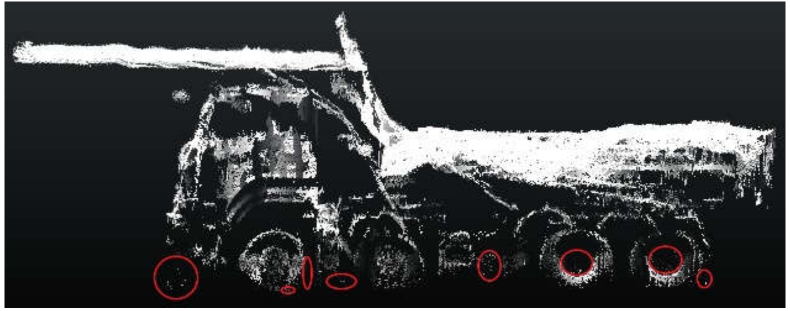
r=0.09m N=763482

r=0.10m N=763899

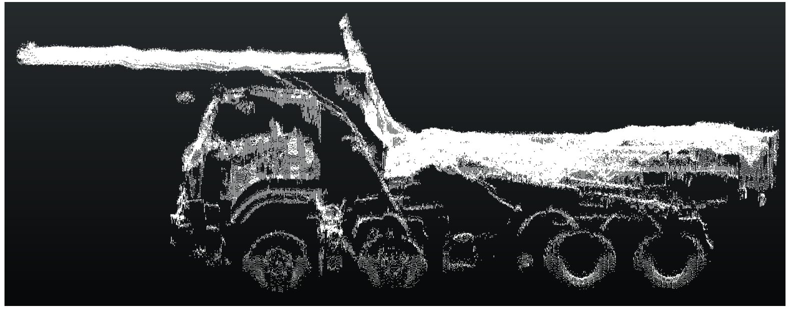
r=0.15m N=764705

不同大小半径r的去噪结果

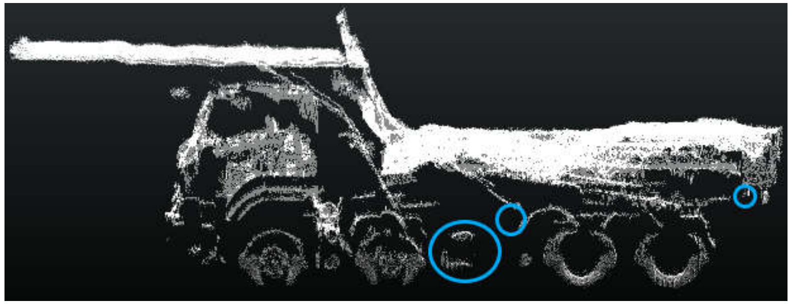
货车点云数据处理



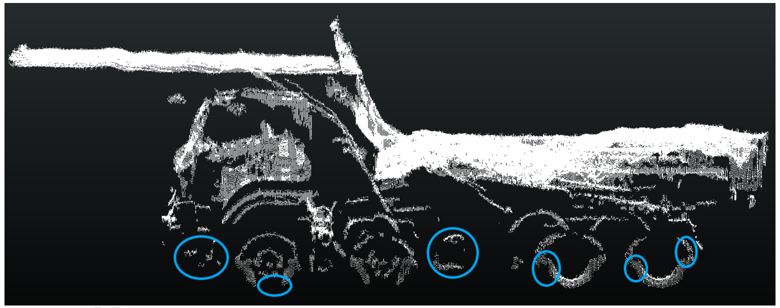
$N_{threshold}$=5 N=764745



$N_{threshold}$=15 N=763482

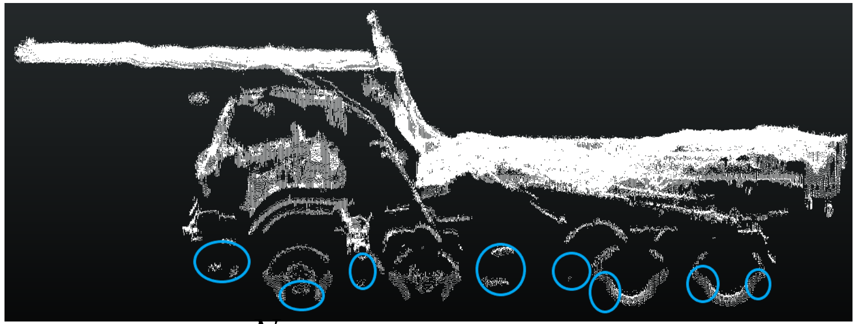


$N_{threshold}$ =20 N=762784



$N_{threshold}$ =30 N=760689

$r = 0.09\text{m}$



$N_{threshold}$ =50 N=754724

当阈值 $N_{threshold}$ 较小，红色圆圈标注的离散点未被滤除,当阈值 $N_{threshold}$ 过大，存在数据缺失

货车点云数据处理

$$cell(l,w,h)$$

设置体素单元

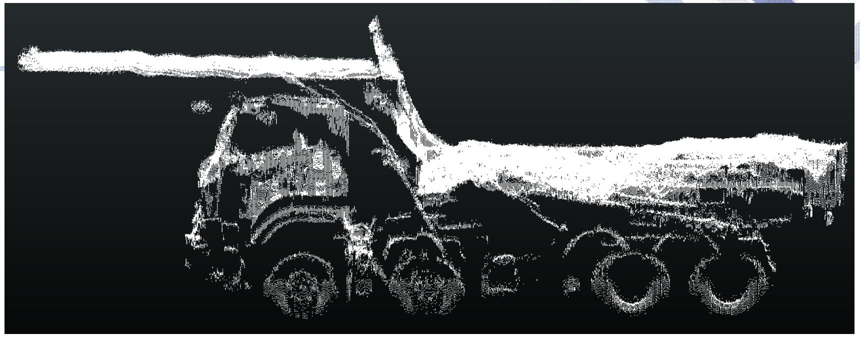
$$\begin{cases} i = \text{int}(\frac{x}{l}) \\ j = \text{int}(\frac{y}{w}) \\ k = \text{int}(\frac{z}{h}) \end{cases}$$

编号

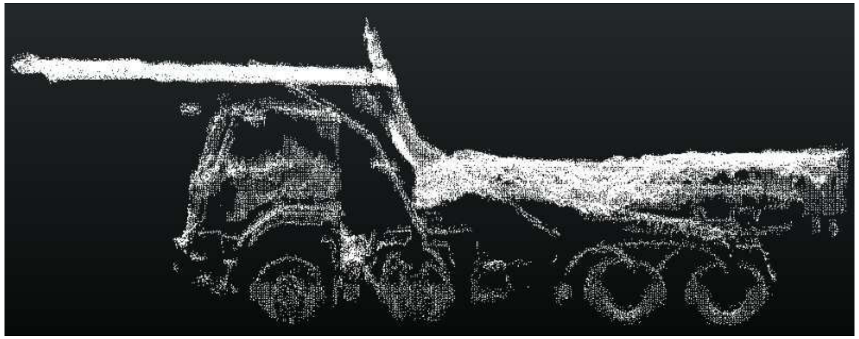
$$c_{ijk} = \frac{1}{n}\sum_{m=1}^{n} p_m$$

精简

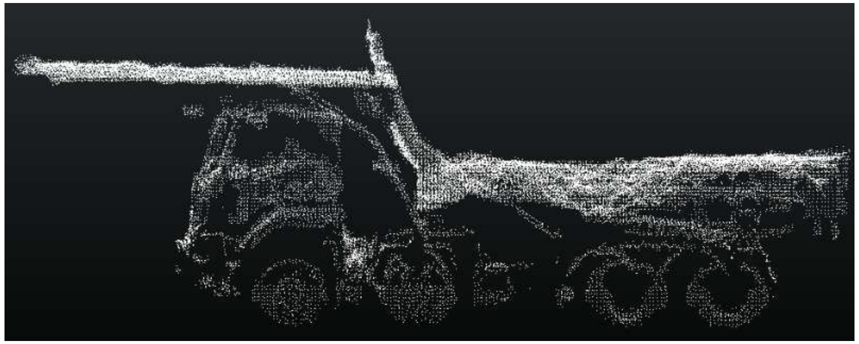
体素滤波器主要功能可以在不破坏点云几何结构功能的前提下，减少点得到数量；去除一定程度的噪音点和离群点。



滤波前 763482



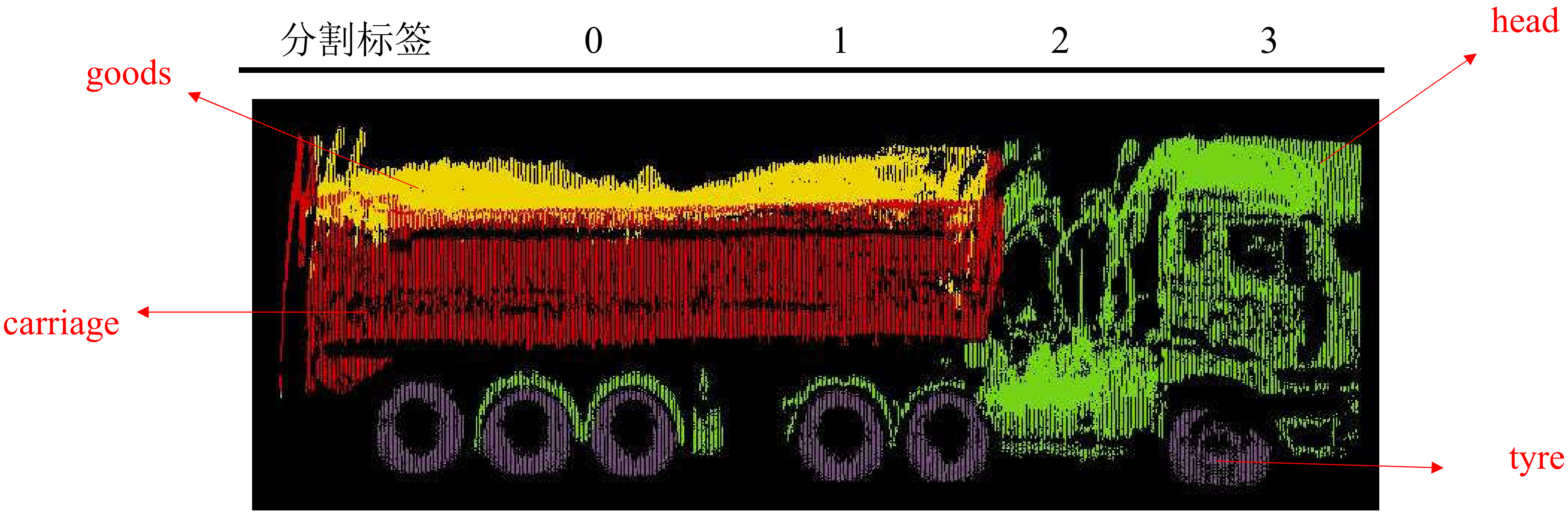
cell=0.03m 96547



cell=0.05m 37534

货车点云标注

	head	carriage	goods	tyre
分割标签	0	1	2	3


goods
head
carriage
tyre

1、数据集存放：数据预处理以及数据标注的点云数据共300帧，每帧分别以标签为命名存放于同一文件夹下，如第i帧货车数据，文件下标签点云的命名为head_i，carriage_i，goods_i，tyre_i四种类别。

2、数据格式：npy格式具有数据加载更加快速以及占用更少的储存空间的优势，标签点云文件转换为npy文件中进行下一步训练。

基于点云数据处理的货车分割

货车分割任务和点云技术结合，利用深度学习的方法快速处理获取货车的数据，可以高速识别点云信息从而实现分割，实现货车分割任务的自动化和高精确化。

基于点云数据处理的货车分割

点处理网络

Step 1

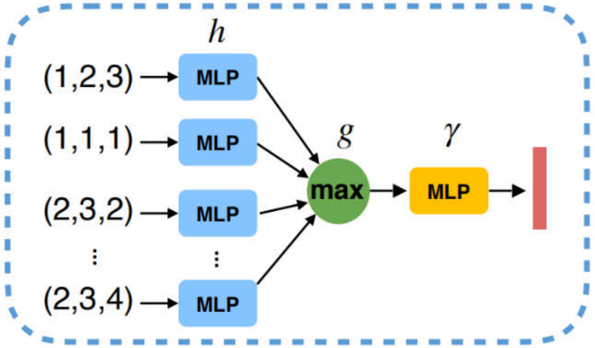
货车分割网络

Step 2

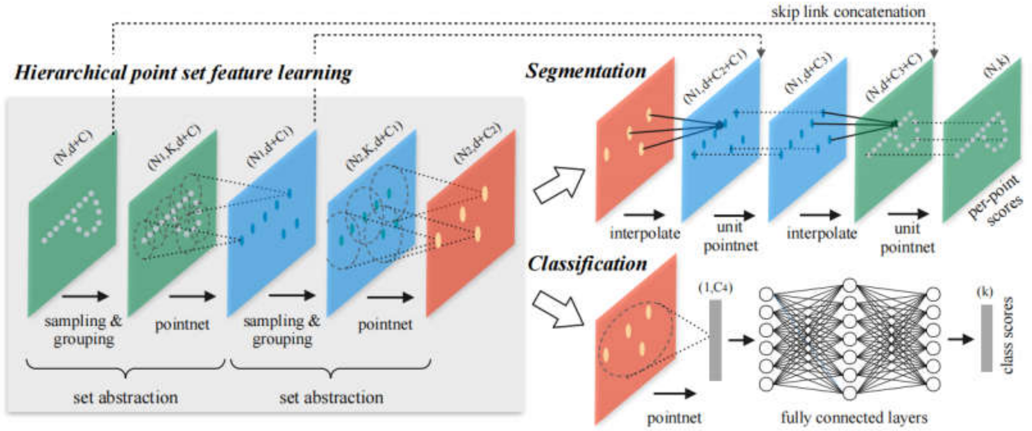
实验分析

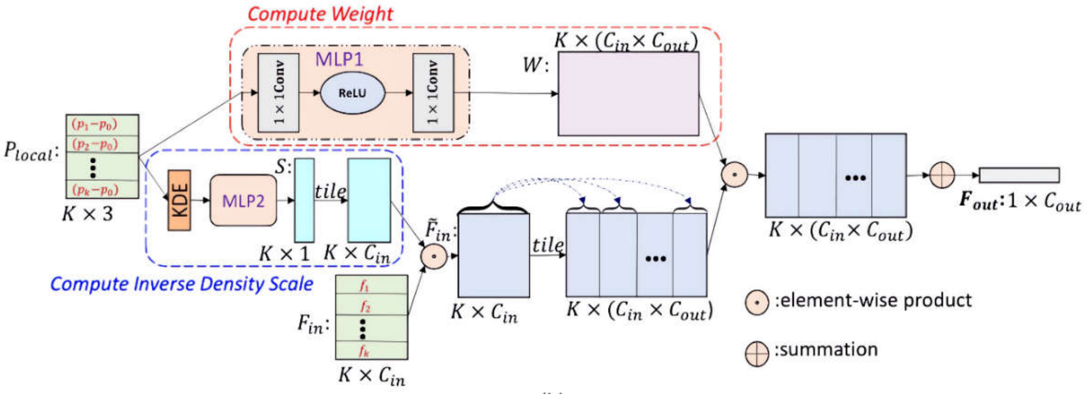
Step 3

基于Point-wise的点云分割网络


h
(1,2,3)
(1,1,1)
(2,3,2)
(2,3,4)
MLP
g
max
γ
MLP

$$f\left(x_1,x_2,x_3,...,x_n\right)=\gamma\cdot g(h(x_1),h(x_2),h(x_3),...h(x_n))$$


skip link concatenation
Hierarchical point set feature learning
(N,d+C)
(N₁,K,d+C)
(N₁,d+C₁)
(N₂,K,d+C₁)
(N₂,d+C₂)
sampling & grouping
pointnet
sampling & grouping
pointnet
set abstraction
set abstraction
Segmentation
(N₁,d+C₂+C₁)
(N₁,d+C₃)
(N,d+C₃+C)
(N,k)
interpolate
unit pointnet
interpolate
unit pointnet
per-point scores
Classification
(1,C₄)
(k)
pointnet
fully connected layers
class scores


Compute Weight
MLP1
1 × 1Conv
ReLU
1 × 1Conv
K × (Cin × Cout)
W:
Plocal:
(p₁−p₀)
(p₂−p₀)
(pₖ−p₀)
K × 3
KDE
MLP2
S:
tile
K × 1
K × Cin
Compute Inverse Density Scale
F̃in:
tile
K × Cin
K × (Cin × Cout)
K × (Cin × Cout)
Fout:1 × Cout
Fin:
f₁
f₂
fₖ
K × Cin
⊙ :element-wise product
⊕ :summation

1、点表征，sampling+grouping整合局部邻域

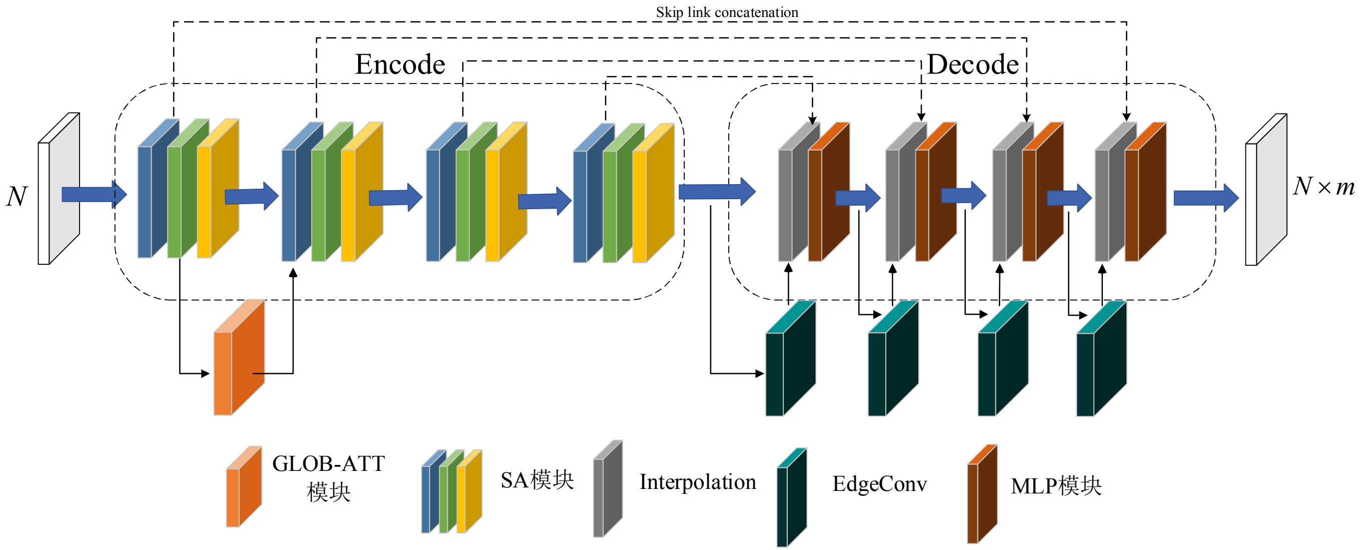
2、层次化特征学习

3、分割任务encoder-decoder结构，对应层 local-global feature拼接

4、非均匀密度，MSG(Multi-scale grouping) MRG(Multi-resolution grouping)

$$PointConv(S,W,F)_{xyz}=\sum_{(\delta x,\delta y,\delta z)\in G}S(\delta x,\delta y,\delta z)W(\delta x,\delta y,\delta z)F(x+\delta x,y+\delta y,z+\delta z)$$

货车点云分割网络


Skip link concatenation
Encode
Decode
N
N × m
GLOB-ATT
模块
SA模块
Interpolation
EdgeConv
MLP模块

GLOB-attetion

全局注意力掩码

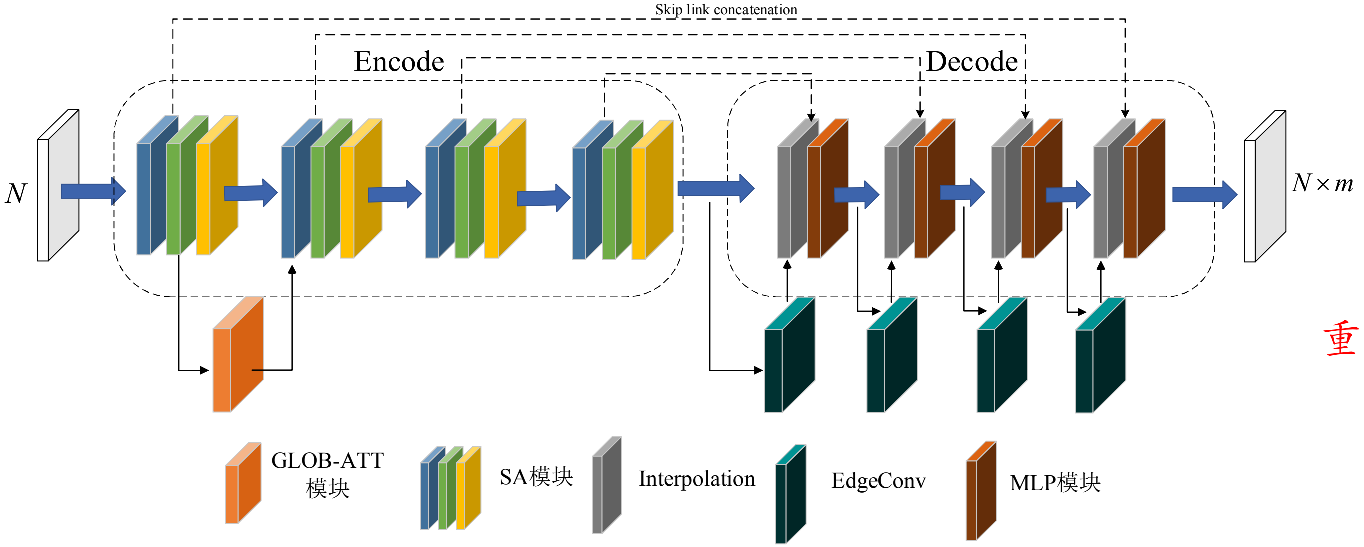
特征提取模块

Set Abstraction模块

特征传输模块

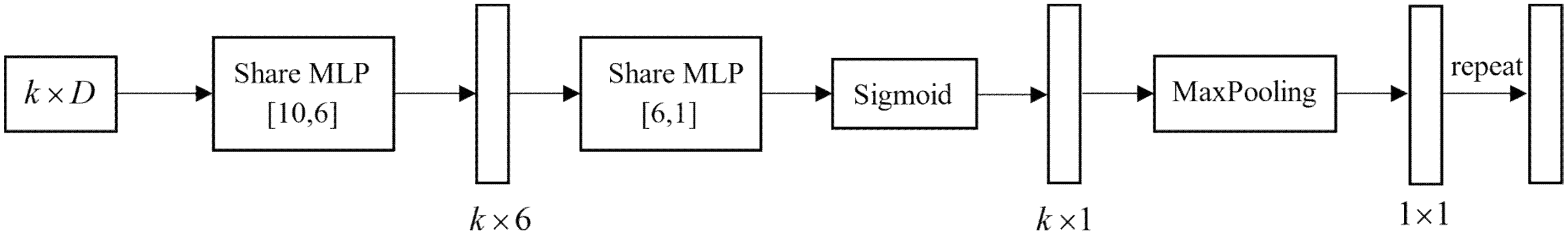
反距离上采样+Edgeconv

全局注意力掩码

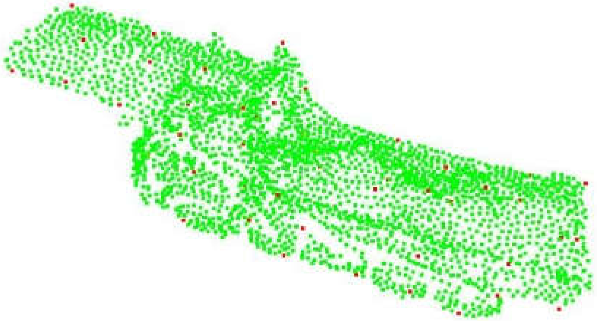

Skip link concatenation
Encode
Decode
N
$N\times m$
GLOB-ATT
模块
SA模块
Interpolation
EdgeConv
MLP模块

重要性：

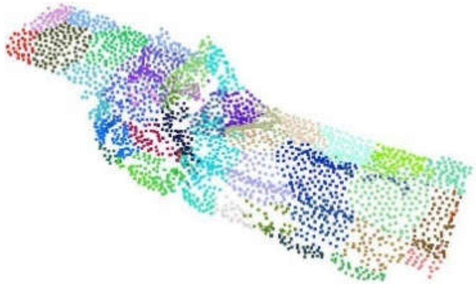
$$x_i^{GA} = \max_{j\in[1,k]} Sigmoid(h_{\Theta_2}(x_i^{SG})) \in R^{1\times1}$$


$k\times D$
Share MLP
[10,6]
$k\times 6$
Share MLP
[6,1]
Sigmoid
$k\times 1$
MaxPooling
1×1
repeat

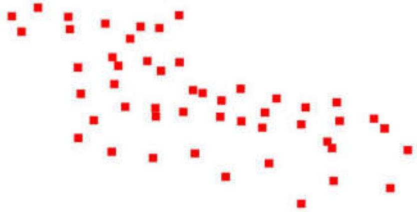
特征提取模块



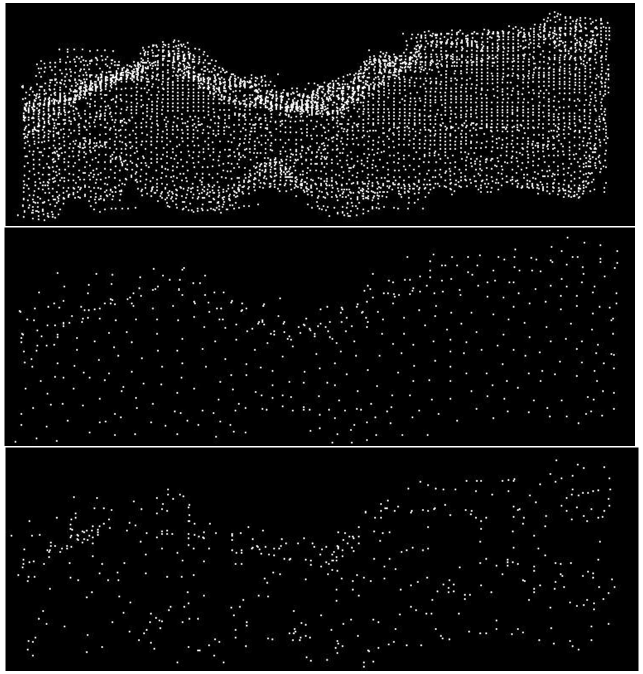
采样层



分层组

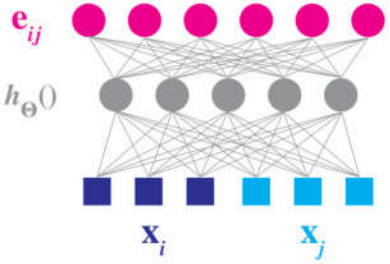


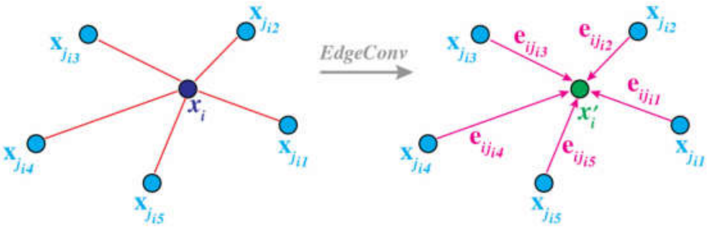
特征提取层



$$f_{GE}(\{\mathbf{F}_i^{(k)}\}_{i=1}^{N_0}) = \gamma(\mathrm{concat}(\{\max_{1,2\ldots k}(h(\mathbf{F}_i^{(k)})) \mid i \in \{1,2,\cdots,N_0\}\}))$$

特征传输模块


$\mathbf{e}_{ij}$
$h_\Theta()$
$\mathbf{x}_i$
$\mathbf{x}_j$


$\mathbf{x}_{ji2}$
$\mathbf{x}_{ji3}$
$\mathbf{x}_i$
$\mathbf{x}_{ji1}$
$\mathbf{x}_{ji4}$
$\mathbf{x}_{ji5}$
EdgeConv
$\mathbf{x}_{ji3}$
$\mathbf{e}_{iji3}$
$\mathbf{e}_{iji2}$
$\mathbf{x}_{ji2}$
$\mathbf{e}_{iji1}$
$\mathbf{x}_i'$
$\mathbf{x}_{ji1}$
$\mathbf{e}_{iji4}$
$\mathbf{e}_{iji5}$
$\mathbf{x}_{ji4}$
$\mathbf{x}_{ji5}$

平移不变性

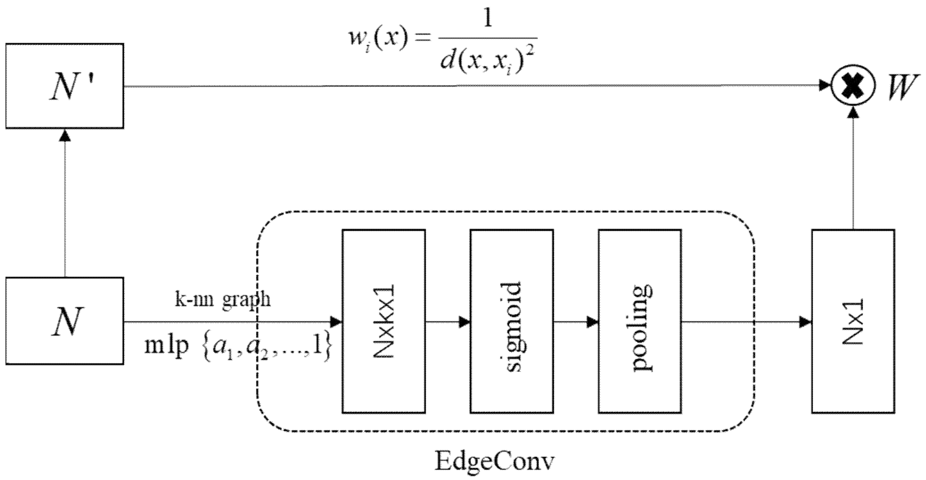
$$\begin{aligned} e_{ijm}' &= \mathrm{Re}LU(\theta_m \cdot (x_j + T - (x_i + T)) + \phi_m \cdot (x_i + T)) \\ &= \mathrm{Re}LU(\theta_m \cdot (x_j - x_i) + \phi_m \cdot (x_i + T)) \end{aligned}$$

$$h_\Theta(x_i, x_j) = \overline{h}_\Theta(x_i, x_j - x_i)$$

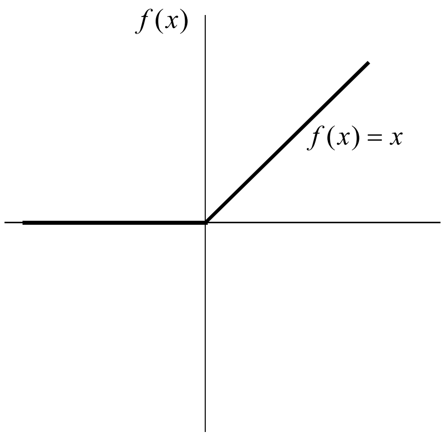
$$e_{ijm}' = \mathrm{Re}LU(\theta_m \cdot (x_j - x_i) + \phi_m \cdot x_i)$$

$$a_i = \max_{j:(i,j)\in E_i} \{sigmoid(e_{ij}')\}$$

$$f^{(j)}(x) = \frac{a\sum_{i=1}^{k} w_i(x) f_i^{(j)}}{\sum_{i=1}^{k} w_i(x)} \quad , \ j = 1, \dots, C$$

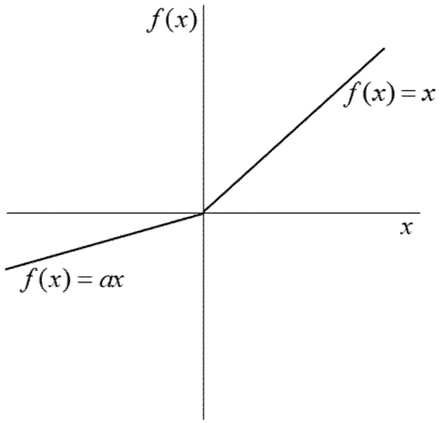

$w_i(x) = \frac{1}{d(x, x_i)^2}$
N'
W
N
k-nn graph
mlp $\{a_1, a_2, \dots, 1\}$
Nxkx1
sigmoid
pooling
Nx1
EdgeConv

激活函数



$$f(x)=\max(0,x)$$

ReLU



$$f(x)=\max(ax,x)$$

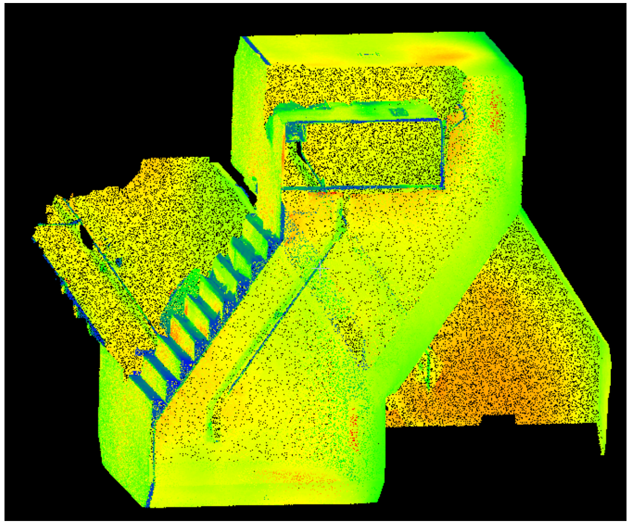
LeakyReLU

所使用LeakyReLU激活函数，可以在反向传播过程中，对于LeakyReLU激活函数输入小于零的部分，不同于ReLU也可以计算得到梯度，避免了梯度方向锯齿问题。

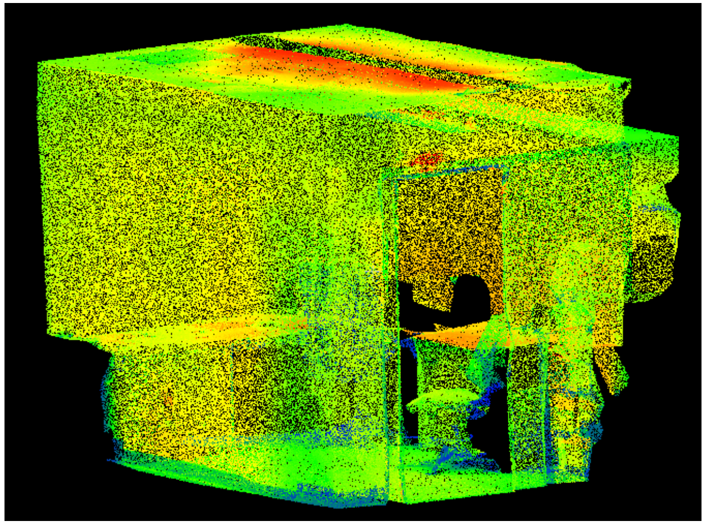
数据集简介

1、所采集货车点云数据集，共300帧，包含3D点云及不同部位标注，如head、tyre、carriage、goods，包含坐标值（XYZ）以及颜色信息（RGB）。

2、S3DIS（Stanford Large-Scale 3D Indoor Spaces）点云数据集，用于研究三维室内场景分割问题。11个场景中包括如天花板、地板、墙壁、梁、柱、窗户、门、桌子、椅子、沙发、书柜、板以及其他等13个语义类别，每个数据点包括点云坐标值（XYZ）以及颜色信息（RGB）。



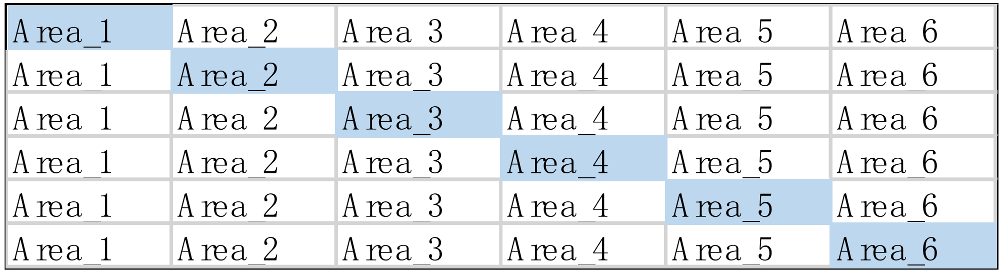
(a) Hallway



(b) Office

评价方法、指标及配置

1、六折交叉验证



	Area_1	Area_2	Area_3	Area_4	Area_5	Area_6
	Area_1	Area_2	Area_3	Area_4	Area_5	Area_6
	Area_1	Area_2	Area_3	Area_4	Area_5	Area_6
	Area_1	Area_2	Area_3	Area_4	Area_5	Area_6
Best!	Area_1	Area_2	Area_3	Area_4	Area_5	Area_6
	Area_1	Area_2	Area_3	Area_4	Area_5	Area_6

2、指标

$$OA=\frac{1}{N}\sum_{C} TP_{\mathrm{i}}$$

总体分割准确度：所有正确预测的样本/样本总数，代表全局的准确性

$$IoU=\frac{TP}{TP+FP+FN}$$

交并比：某一类预测结果和真实值的交集与并集的比值

$$mIoU=\frac{1}{C+1}\sum_{i=0}^{C} IoU$$

平均交并比：综合考虑所有类别性能的指标

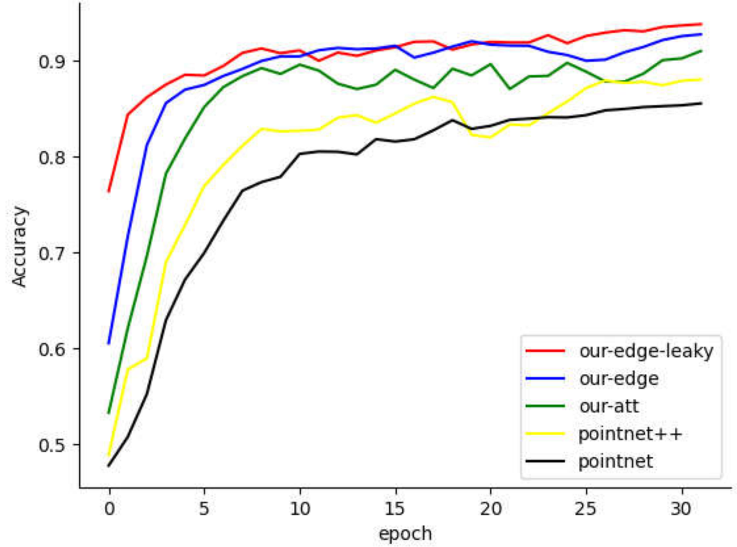
3、配置

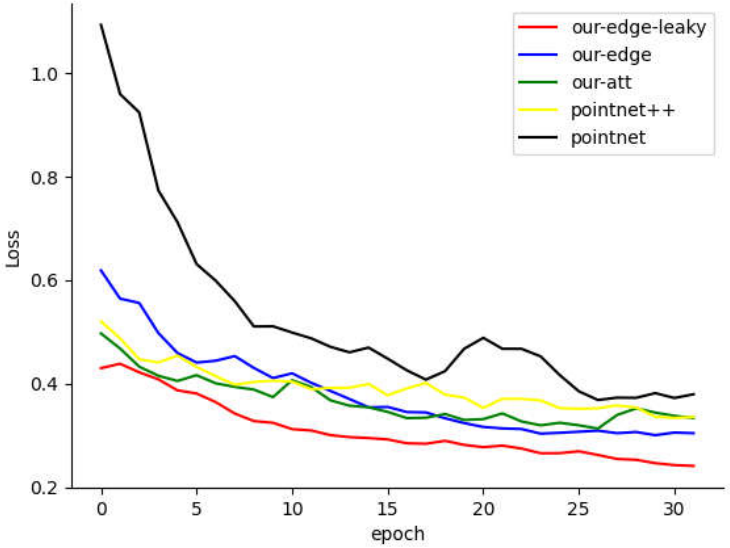
实验配置	型号/版本
CPU	8核Inter CORE i7-10700
显卡	Nvidia GTX 1660
内存	16G
操作系统	Ubuntu20.04
CUDA	10.2
CUDNN	7.6.5

指标	参数
Batch_size	16
学习率	0.01
优化率	Adam
npoints	4096
激活函数	LeakyReLU
衰减速率	0.4

货车点云数据实验结果

准确率



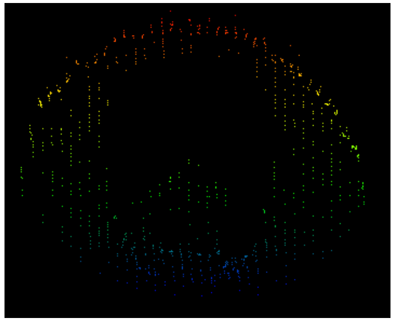


损失率

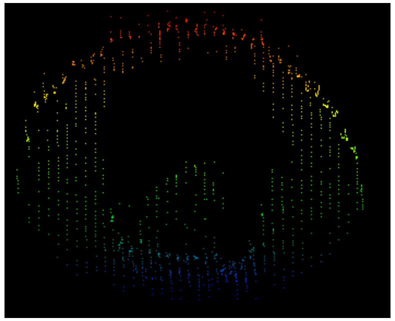
	Head	Carriage	Goods	Tyre	mIoU	oAcc
PointNet	84.9	53.4	84.2	59.5	56.3	88.7
PointNet++	84.2	63.8	86.8	59.2	58.8	90.0
PointCNN	80.7	71.6	82.2	74.3	61.8	88.1
RanLA-Net	85.8	65.3	89.9	82.0	64.5	91.6
Our-Att	85.1	68.5	89.1	78.9	64.3	91.3
Our-Edge	88.5	73.4	89.5	86.1	67.5	92.9
Our	91.6	74.6	90.9	85.3	68.5	93.9

总体精度可达到93.9%，而mIoU提升至68.5%，优于经典分割网络PointNet和PointNet++，其中，mIoU相较PointNet、PoinNet++高出12.2%、9.7%，oAcc提高了5.2%、3.9%。

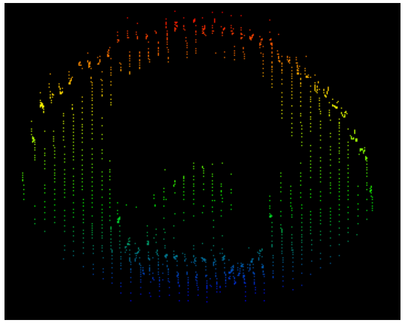
鲁棒性实验



40%

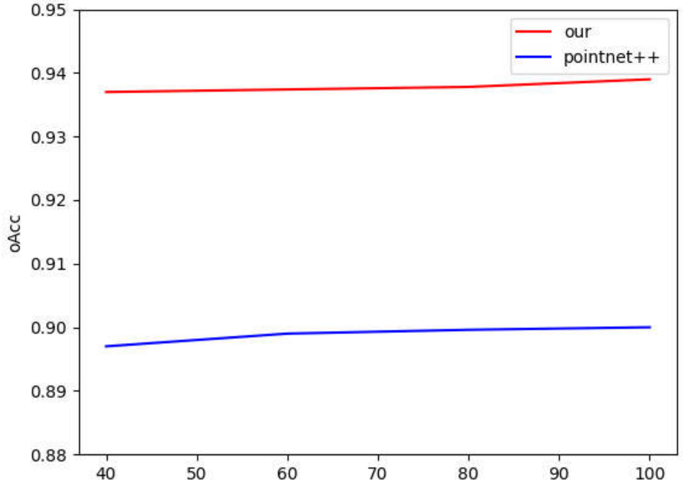


60%



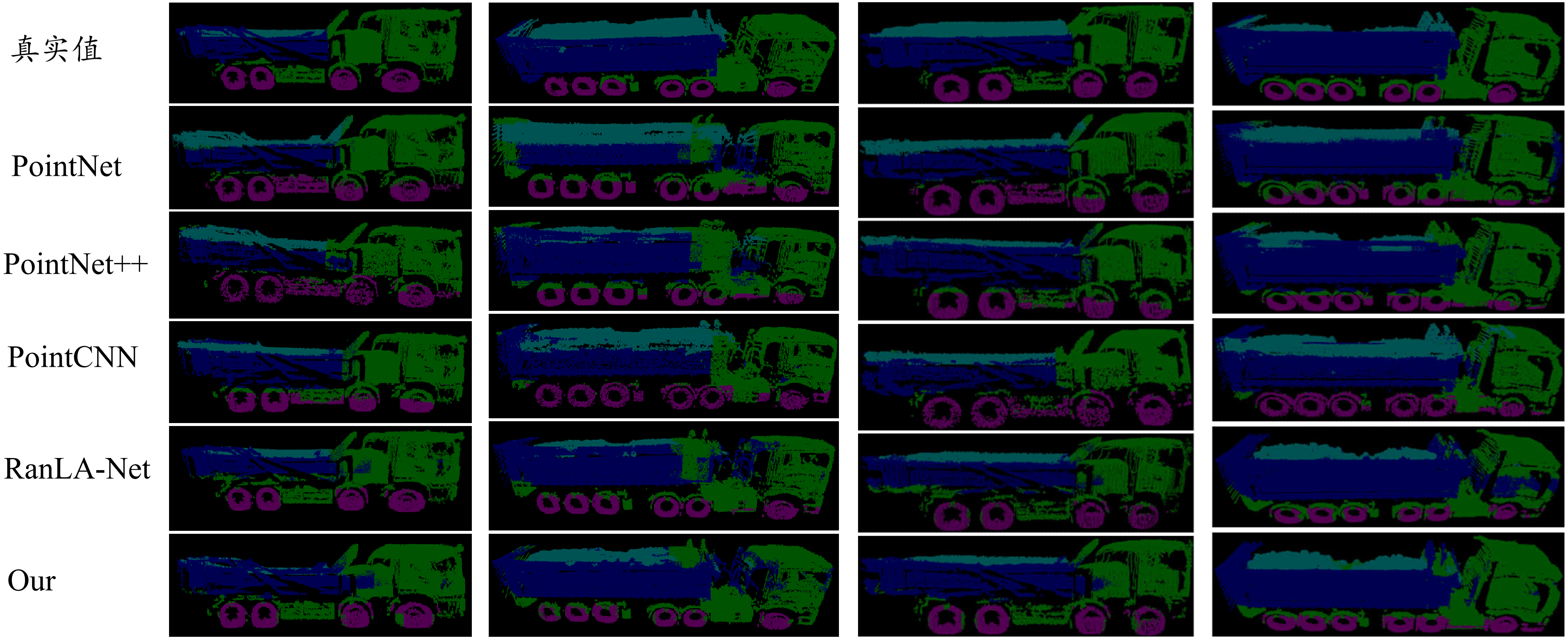
80%

验证本章网络对点云密度和丢失数据信息的鲁棒性



使用PCL库中的随机下采样算法分别采样原始点云数据的40%、60%、80%数据作为输入，分别对本章设计网络以及PointNet++进行预测。

分割结果展示


真实值
PointNet
PointNet++
PointCNN
RanLA-Net
Our

分割结果展示

真实值

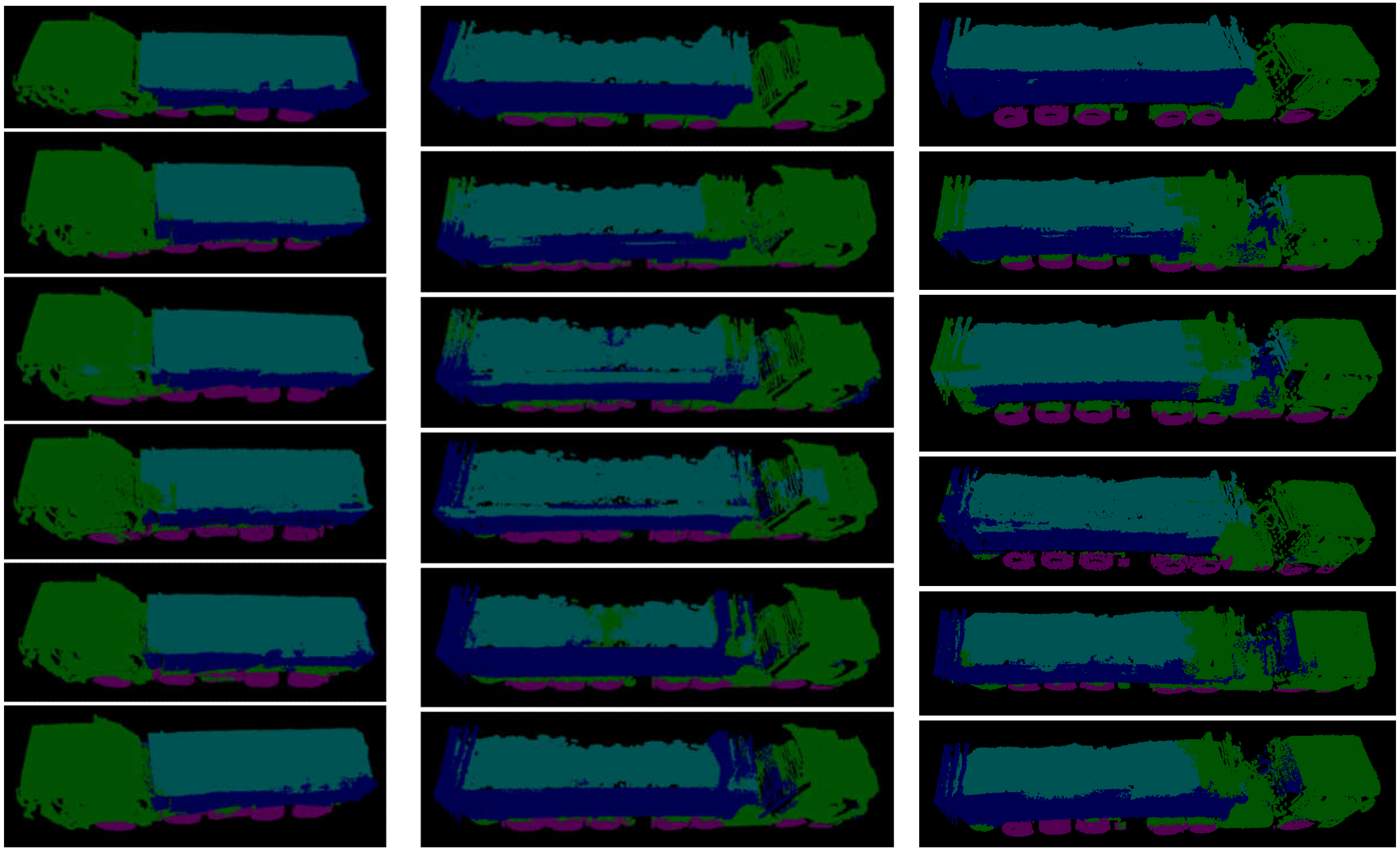
PointNet

PointNet++

PointCNN

RanLA-Net

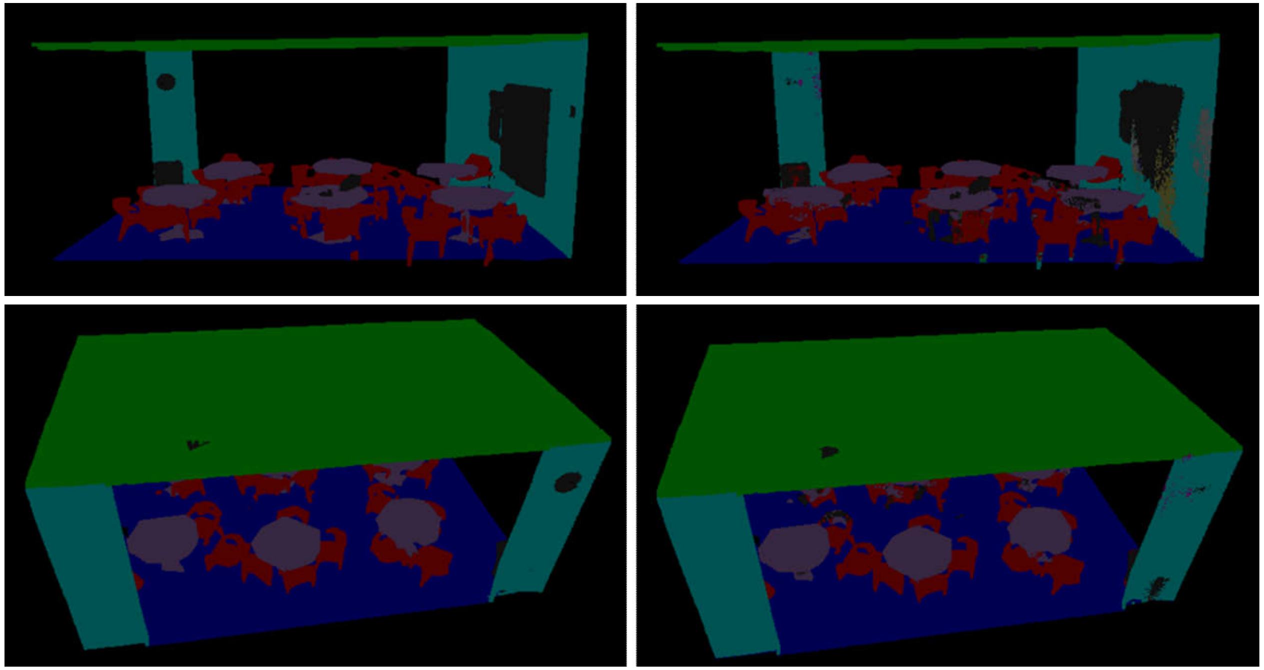
Our



S3DIS数据集实验结果

IoU	Ceil	Floor	Wall	Win.	Door	Tabel	Chair	Case	Board	Clutter	mIoU	oAc
PointNet	87.4	97.8	71.2	52.1	16.3	58.2	48.6	48.3	39.0	36.2	43.6	78.9
PointNet++	89.4	97.7	75.4	58.3	19.5	69.2	79.0	59.1	58.7	41.6	53.5	82.9
Our	91.0	97.5	77.5	63.3	20.9	71.5	78.9	62.7	52.5	46.2	55.2	84.5

大堂分割结果展示（左：真实值，右：预测值）



曲面重建及体积计算

本章中使用分割的货物点云数据实现体积计算。对不规则曲面的货物点云进行体积计算包括两种方法，对货物进行曲面重建进而实现体积计算以及基于货物的点云数据计算体积。

曲面重构算法

Poisson重构算法

隐式曲面重建方案；从能量构建等式方程→离散化基函数→基函数系数

平滑滤波卷积指示函数，避免直接计算出的向量场在表面边缘处的无限大值，获得更准确的梯度场。

Marching Cubes重构算法

(1)定义立方体基于保罗·伯克约定进行编号；(2)索引立方体顶点以及边；

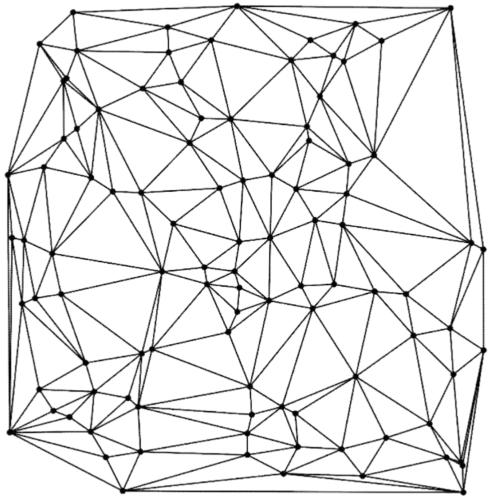
(3)线性插值计算立方体边缘交点　$P = P_1 + (isovalue - V_1)(P_2 - P_1)/(V_2 - V_1)$

Greedy Projection Triangulation重构算法

快速三角化；基于增量表面增长原理

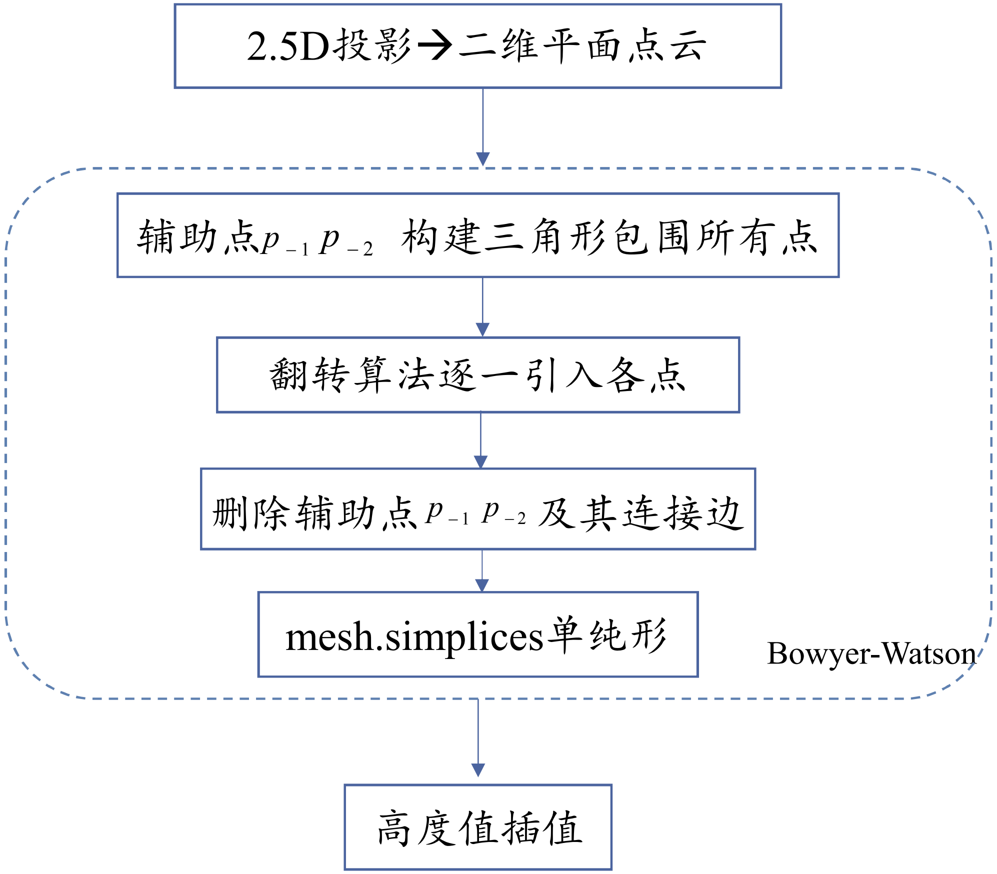
(1)Kd-tree最近邻搜索；(2)使用切平面进行领域投影，并排序；(3)通过可见性和距离等标准进行删减。

2.5D Delaunay重构算法



Delaunay三角剖分

2.5D场景是指一个由二维平面和一个高度值（或深度值）组成的场景，在2.5D场景中，每个点都有X、Y坐标和一个高度值（或深度值）。相较三维场景，2.5D场景在重建方面简单、数据量小。



2.5D投影→二维平面点云

辅助点p_{-1} p_{-2} 构建三角形包围所有点

翻转算法逐一引入各点

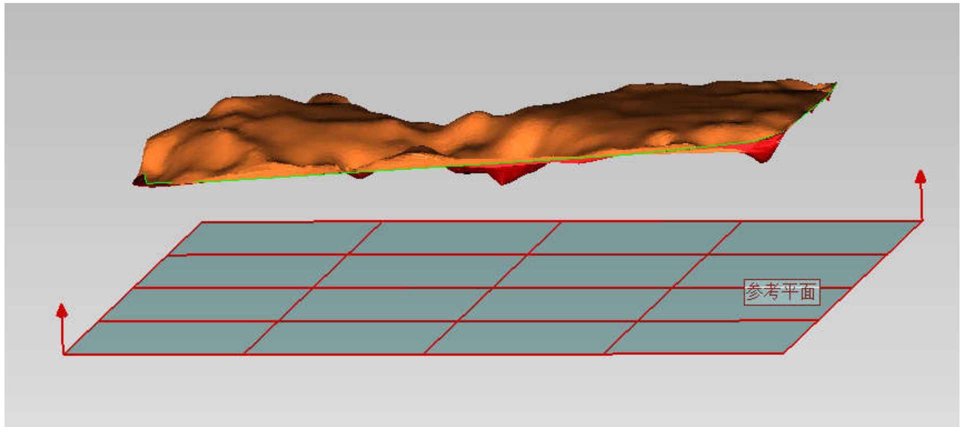
删除辅助点p_{-1} p_{-2} 及其连接边

mesh.simplices单纯形

Bowyer-Watson

高度值插值

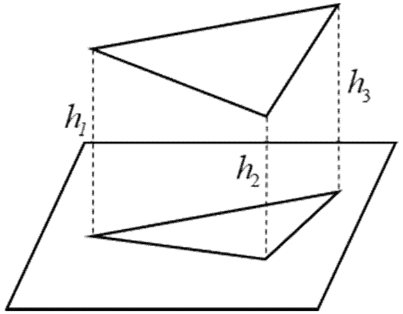
三角网法



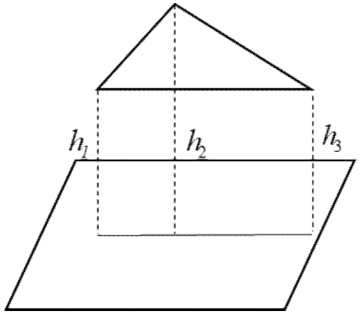
计算重构模型与基准面之间的体积

对重构模型中每个三角形与基准面之间体积进行计算

$$S_{投} = \frac{1}{2}\left[(x_1 - x_3)(y_2 - y_3) - (x_2 - x_3)(y_2 - y_3)\right]$$

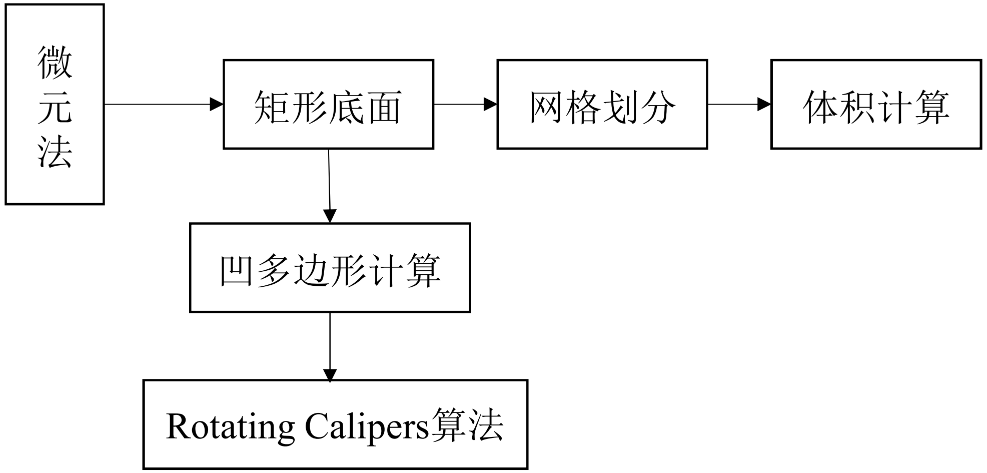


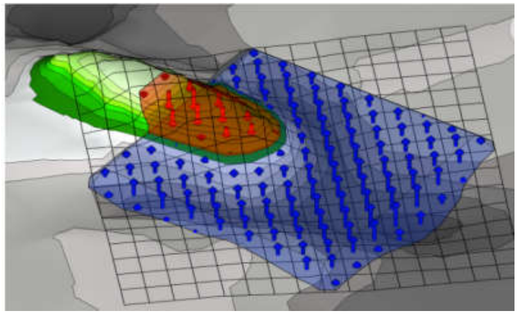
$$V = \frac{1}{3} S_{投} (h_1 + h_2 + h_3)$$



$$V = 0$$

微元法


微元法
矩形底面
网格划分
体积计算
凹多边形计算
Rotating Calipers算法



$$V_{Grid} = \frac{1}{k}\sum_{i=1}^{k} q_i \times g^2$$

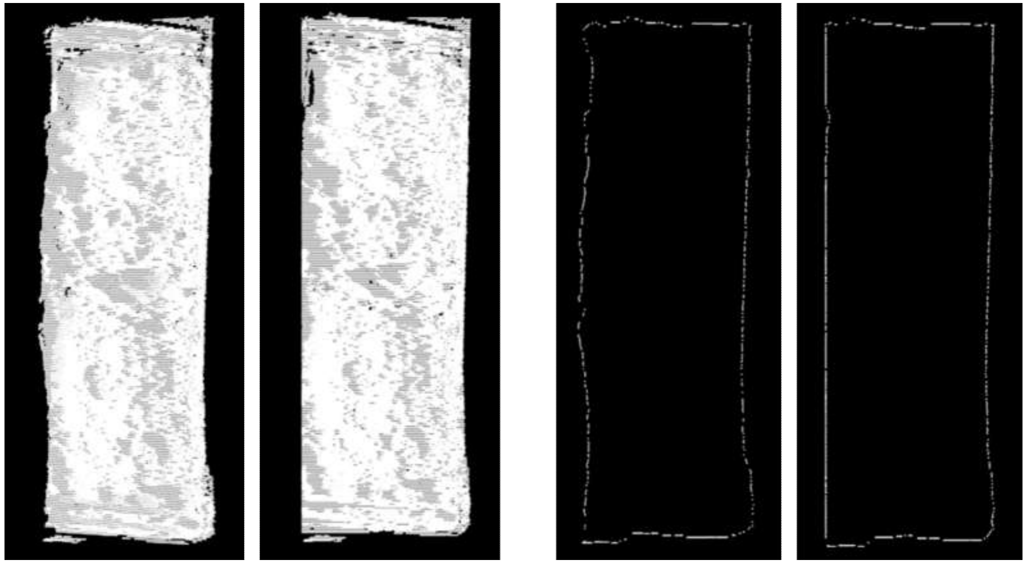
确定点云货物的矩形单元及平面。使用凹包多边形计算以及Rotating Calipers算法得到计算体积货物的矩形底面

q_i代表网格中的点数据相较基准平面高度，k个点数据，g为网格边长。

凹多边形计算算法

算法 4.1：货物的凹多边形计算算法

```
输入：一个点云集 $P$，包含 N 个点，滤波范围 $[z_{\min}, z_{\max}]$，投影阈值 $D_{thredshold}$
输出：凹多边形点云集 $P_{concave}$
1．初始化 pcl::PointCloud<pcl::PointXYZ> cloud_filtered,cloud_projected,concavehull
2．构建 PassThrough 滤波器，参数为 $[z_{\min}, z_{\max}]$，得到 cloud_filtered 滤波
3．创建 ModelCoefficients 投影平面 $ax+by+cz+d=0$
4．初始化 ProjectInliers 对象 proj
5．SetModelType(pcl::SACMODEL_PLANE) 对象 proj 参数为平面
6．proj.cloud_filtered $\rightarrow$ cloud_projected
7．初始化 ConcaveHull 对象，参数 alpha 设置
8．while alpha>0 do
9．    do 检测凹点 concave_indices
10．       cloud_projected 中凸点 $p_i$ 为圆心，以 alpha 半径为邻域点集 $n$
11．       由小到大的极角 $\theta=\{\theta_1 \ldots \theta_j\}$ 排列 $\{n_1 \ldots n_j\}$
12．       if $\alpha_{n_{\lceil \frac{j}{2} \rceil -1} n_{\lceil \frac{j}{2} \rceil +1}} > 180°$ then
13．           concave_indices+= $n_{\lceil \frac{j}{2} \rceil}$
14．       end
15．       从 $n_{\lceil \frac{j}{2} \rceil}$ 开始重复 10.11.12.13 至凹点不存在
16．    end
17．    if concave_indices==0 then
18．       break
19．    end
20．    alpha-=0.01
21．return concavehull
```



(a) 投影图（左 随机一致 右XOY）　(b) 凹多边形（左 随机一致 右XOY）

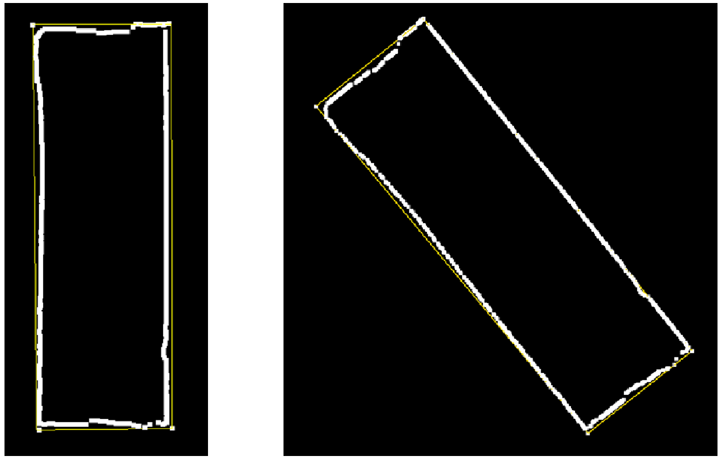
投影平面实验对比

参数	Filter	平面	alpha
取值	[-5.0,-2.0]	[0,0,1,0]	0.01

Rotating Calipers算法

算法 4.2：Rotating Calipers 最小外接矩形算法

```
输入：点集合$\{C\}$
输出：MinAreaRect 最小外接矩形，旋转角度$\gamma$
计算 $C$ 凸包 hull_points=$\{h\}$
初始化 $min\_area = +\infty$
$edge = h[1:] - h[:-1] = \{e_1 ... e_n\}$
for $edge$ 中每个边 $e_i$ do
    计算与 x 轴旋转角度 $\gamma_i$， $R = [[cos(\gamma_i), -sin(\gamma_i)],[sin(\gamma_i), cos(\gamma_i)]]$
    $rot = R \times hull\_points^T$
    计算 $area = (\max(rot_x) - \min(rot_x)) \times (\max(rot_y) - \min(rot_y))$ 面积
    if $area < min\_area$ do
        $MinAreaRect = Rect \times R , min\_area = area , \gamma = \gamma_i$
    end
end
return $MinAreaRect , \gamma$
```

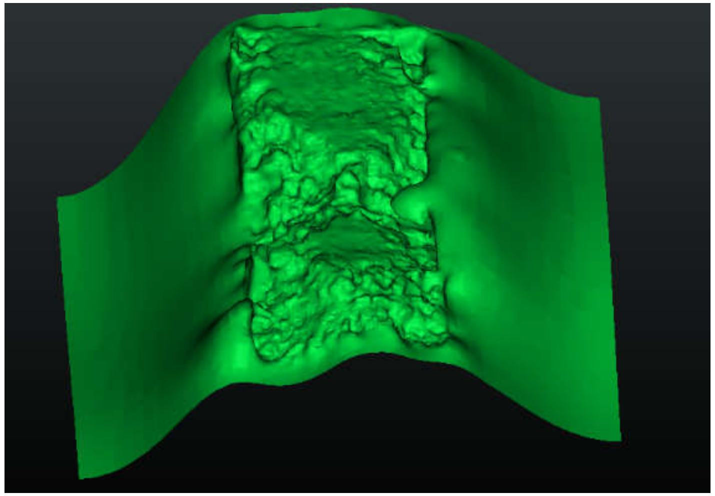


最小外接矩形

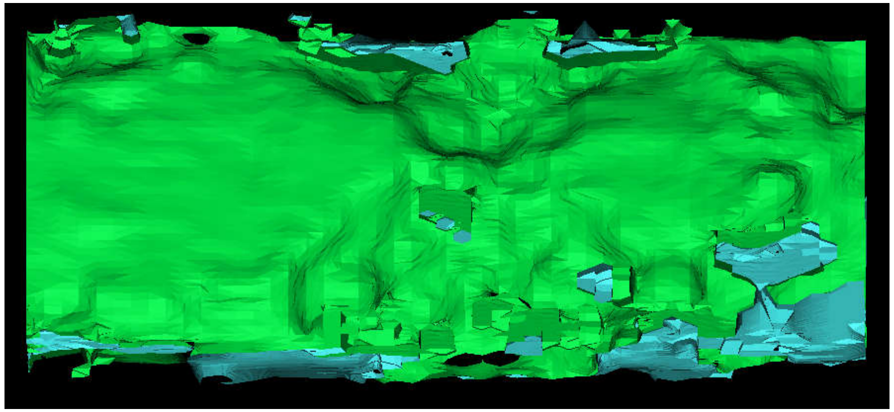
左图为得到的原始凹多边形计算的最小外接矩形结果，黄色框为最小外接矩形，其得到的旋转角度γ为89.98°，基本垂直。右图为旋转45°后得到的最小外接矩形，计算得到旋转角度为44.96°

曲面重构

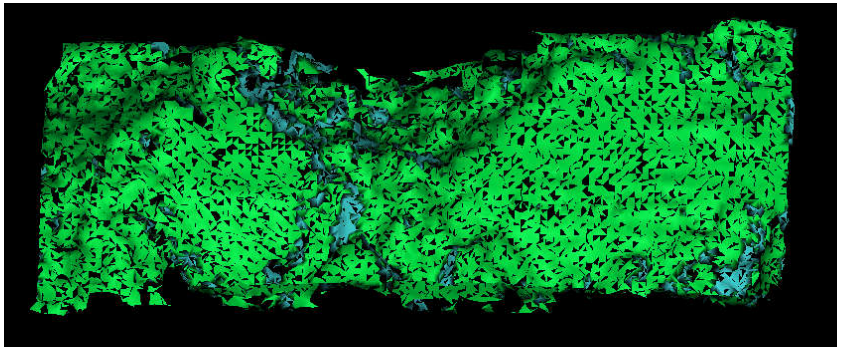
Poission重建

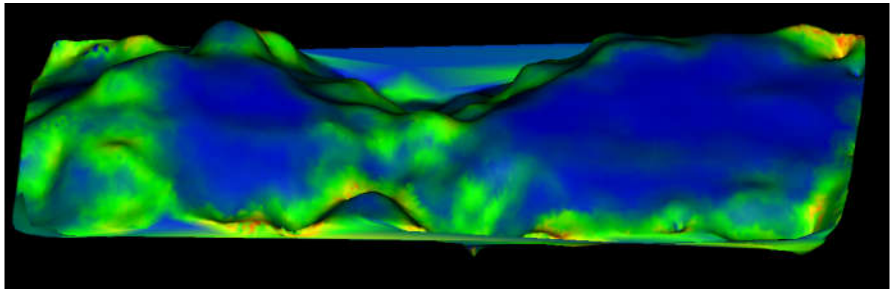


贪婪投影三角化重建



Marching Cubes重建





2.5D Delaunay重建

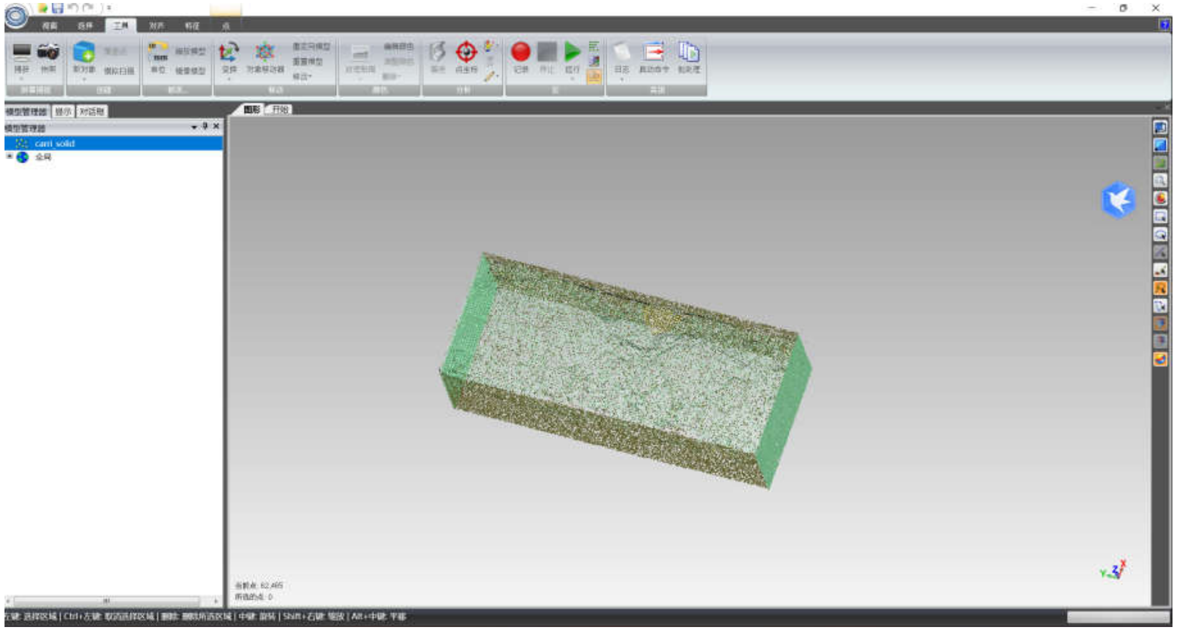
参数	KSearch	Radius	SetMu	Neighbors	MinAngle	MaxAngle	SurfaceAngle
数值	15	0.1	30	100	0°	180°	45°

曲面重构

	Poisson 重建	Maching Cubes 重建	Greedy Projection Triangulation	2.5D Delaunay
$Data_1$	1.020	0.185	0.202	0.049
$Data_2$	0.759	0.336	0.130	0.036
$Data_3$	1.172	0.229	0.205	0.053
$Data_4$	0.737	0.229	0.107	0.030
$Data_5$	1.138	0.187	0.252	0.058
$Data_6$	1.100	0.218	0.224	0.054
$Data_7$	1.342	0.431	0.241	0.058
$Data_8$	1.167	0.304	0.235	0.057
$Data_9$	0.776	0.125	0.138	0.030
$Data_{10}$	0.835	0.161	0.174	0.039

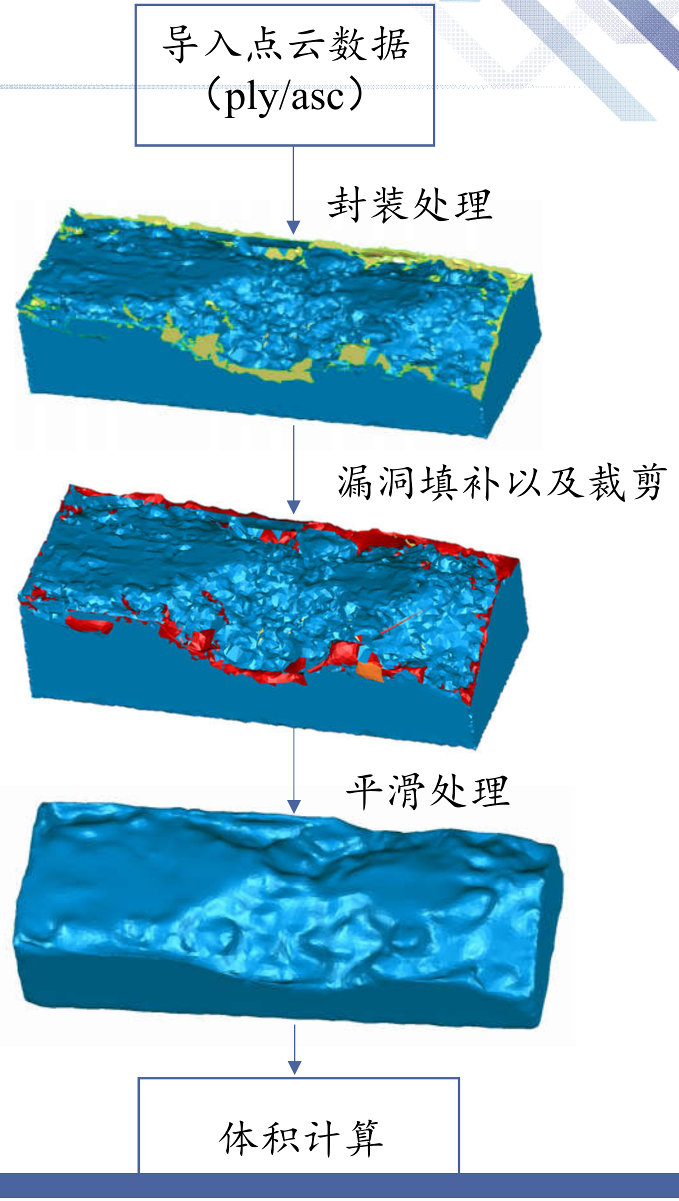
相较其他三个重建算法，2.5D Delaunay重建算法实现曲面任务中有着高效的效率，FPS平均为21.6帧/s。而Greedy Projection Triangulation算法平均每帧为191ms的运行时间。Maching Cubes、Poisson重建算法则具有更大的计算成本。

体积计算-基准软件



ArcGIS、Surfer、飞时达以及Geomagic Studio软件用于体积测量

Geomagic Studio 12是一款强大的三维重建软件，其中包含了许多工具和算法，可用于土方量的体积测量，在园林测量等方面有广泛应用



导入点云数据（ply/asc）

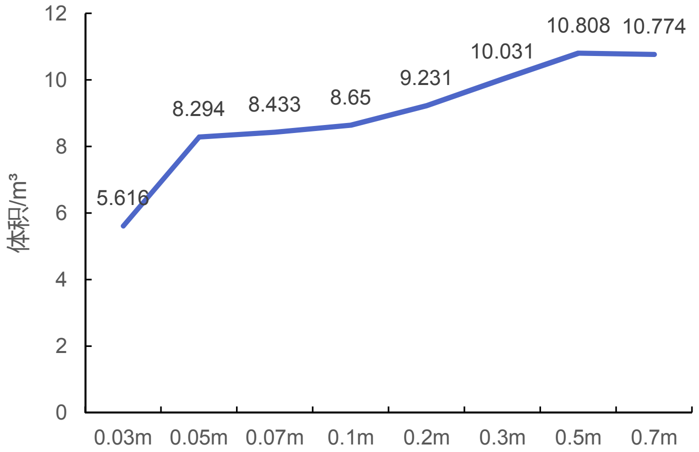
封装处理

漏洞填补以及裁剪

平滑处理

体积计算

体积计算



体积基本与网格呈现正相关关系，当网格越小，会出现更多空洞网格导致计算体积偏小，当网格过大时，会平均覆盖其余点云深度信息，导致计算体积出现误差。

相较Geomagic实际所测的体积为8.78m³，当单元格边长为0.1m大小时体积相差最小。

	$Data_1$	$Data_2$	$Data_3$	$Data_4$	$Data_5$	$Data_6$	$Data_7$	$Data_8$	$Data_9$	$Data_{10}$
DTM	8.851	5.405	9.050	2.502	7.704	7.300	8.415	7.620	3.601	7.401
微元法	8.650	5.122	9.002	2.680	7.689	7.259	8.370	7.323	3.856	6.965
Geomagic	8.782	5.358	9.125	2.390	7.520	7.290	8.420	7.560	3.556	7.330

微元法过于依赖Grid大小且存在部分空洞导致获得的体积变换较不稳定且计算体积偏小，平均误差为0.1933m³，而三角网法利用重建数据得到的体积较为稳定，平均误差为0.0678m³

总结与展望

培养学生综合运用、巩固与扩展所学的基础理论和专业知识，培养学生独立分析、解决实际问题能力、培养学生处理数据和信息的能力。

总结

01 构建工地货车点云数据集

本文主要基于激光雷达实现实际工地场景-过地磅处货车数据采集。

- ✓ 提出了较为新颖的1-2多传感器结合的模式；设计数据拼接模块双雷达的标定流程；进而对垂直驶入的货车结合改进ICP的测长雷达位移算法和数据采集拼接模块的系统进行检测。

02 基于PointNet++改进的货车点云分割研究

- ✓ 提出引入Glob-ATT模块作为点特征的重要性权重参数用于处理采样簇中的特征模式，以突出区域的显著特征,来提高分割的准确性。
- ✓ 结合PointNet++中多尺度的特征来实现特征的提取，并将EdgeConv方法嵌入特征传输模块改进插值系数，从而获取更多的特征信息进行上采样插值。
- ✓ LeakyReLU激活函数

03 构建工地货车点云数据集

- ✓ 实现了基于重建模型和点云原始数据的两种三维体积计算方法

展望

01 构建工地货车点云数据集

- ✓ 行驶货车位移计算

02 基于PointNet++改进的货车点云分割研究

- ✓ 雷达颜色信息的缺失；
- ✓ 对不均衡货车数据分割精度；
- ✓ 加速模型推理

03 构建工地货车点云数据集

- ✓ 重建模型对雷达校准的依赖