# Lidar & 3D目标检测

## 单阶段目标检测主要方法

* 基于点的表示：3DSSD

* 基于Voxel的表示：SECOND、PointPillars、CenterPoint

* 基于投影的表示：YOLO3D，PIXOR、Range Det， RSN

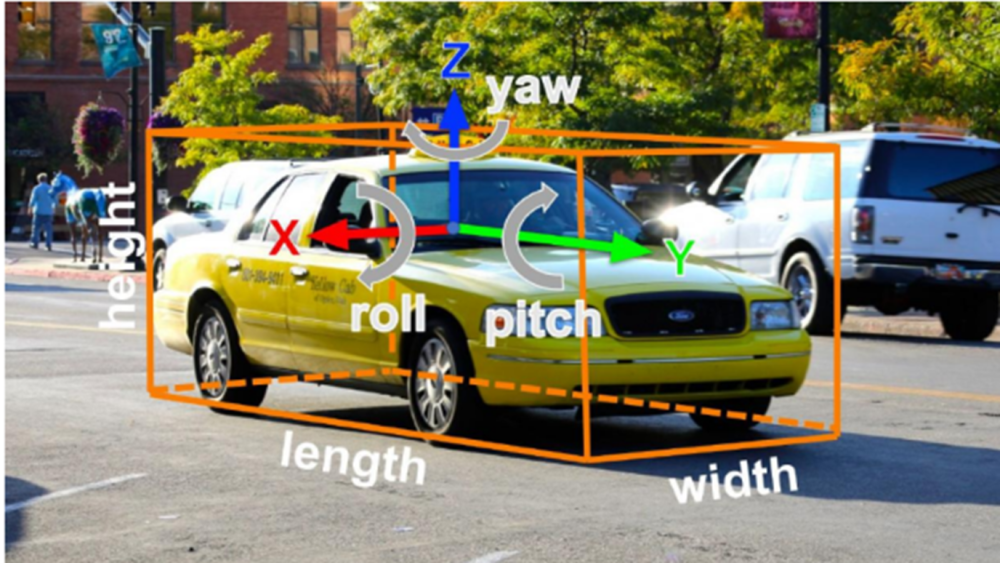
* 基于图的方法：DGCNN

## 多阶段目标检测主要方法

* 基于点云： Point-RCNN、Part-A2、PV-RCNN

* 利用图像：MV3D、Frustum-PointNet、Point-Painting

## 3D目标检测



* 2D目标检测：（x, y, w, h)

* 3D目标检测：（x, y, z, w, h, l, roll, yaw, pitch) 共9个维度

* 在自动驾驶场景中，一般认为汽车行驶在路面上roll角和pitch角几乎为0且对驾驶行为没有影响，因此简化为**（x, y, z, w, h, l, yaw)** 7个维度

* 在BEV视角下，可进一步忽略车辆高度，因此只需要预测**(x, y, h, w, yaw)** 5个维度，仅比2D目标检测多了朝向角

* 3D目标检测的Head部分，与2D目标检测几乎没有太大不同，同样可以使用以下要素

* + anchor based、anchor free

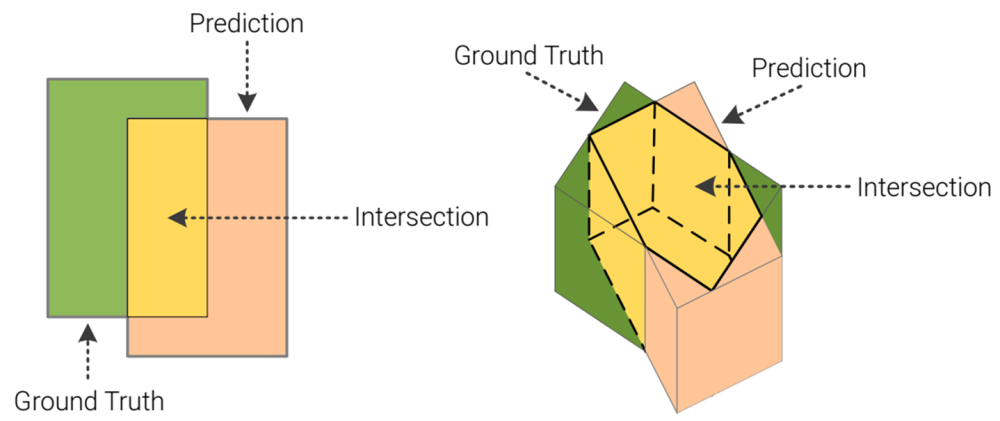
* + 一阶段检测器、二阶段检测器

* + 多分辨率融合（FPN）

* + Smooth L1，Focal Loss，IOU Loss

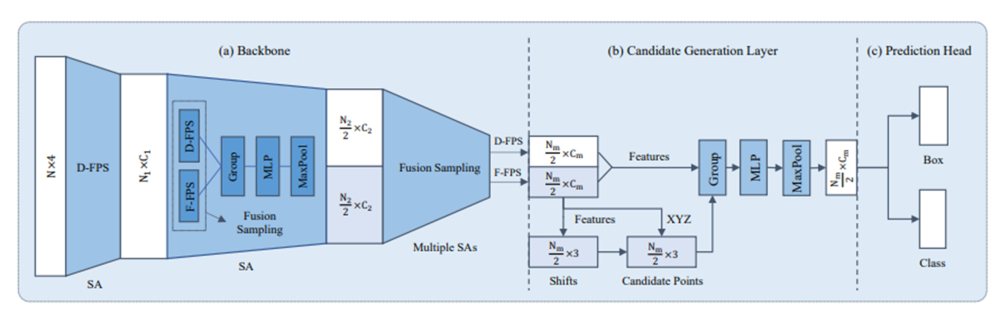
* 自动驾驶场景的3D目标检测，不存在目标重叠的情况，因此不需要复杂的anchor分配机制

* IOU的计算和2D目标检测下略有不同



## 单阶段技术发展

### 3DSSD



基于点的方法，将原始点云作为输入，并根据每个点预测边界框。具体来说，它们由两个阶段组成。在第一阶段，他们首先利用集合抽象 (SA) 层进行下采样和提取上下文特征。之后，特征传播（FP）层被应用于上采样和广播特征到在下采样过程中被丢弃的点。然后应用**3D区域提议网络 (RPN) 来生成以每个点为中心的提议**。基于这些建议，开发了一个细化模块作为第二阶段来给出最终预测。这些方法获得了更好的性能，但它们的推理时间在许多实时系统中通常是无法忍受的。

#### Motivation

当前所有基于点的方法由两个阶段组成，包括提议生成阶段和预测细化阶段。在第一阶段，SA层应用于下采样点以提高效率并扩大感受野，而FP层应用于下采样过程中丢弃点的广播特征以恢复所有点。在第二阶段，一个细化模块优化来自RPN的建议以获得更准确的预测。SA 层是提取点特征所必需的，但FP层和细化模块确实限制了基于点的方法的效率。因此，需要设计轻量级且有效的基于点的单级检测器

然而，移除FP层并非易事，如前所述，主干中的SA层利用D-FPS选择点的子集作为下采样的代表点。在没有 FP层的情况下，框预测网络是在那些幸存的代表点上进行的。尽管如此，这种采样方法只考虑了点之间的相对位置。因此，大部分幸存的代表点实际上是背景点，如地面点，因为其数量很大。换句话说，有几个前景实例通过这个过程被完全擦除，使其无法被检测到

#### F-FPS

为了保留正点（任何实例中的内部点）并删除那些无意义的负点（位于背景上的点），在**采样过程中不仅要考虑空间距离，还要考虑每个点的语义信息**。然而，仅以语义特征距离作为唯一标准会在同一实例中保留许多点，这也会引入冗余。例如，给定一辆汽车，窗户和车轮周围的点的特征之间存在很大差异。因此，将采样这两个部分周围的点，而任一部分中的任何点都可以为回归提供信息。为了减少冗余并增加多样性，**同时应用空间距离和语义特征距离作为FPS的标准**。



#### 融合采样

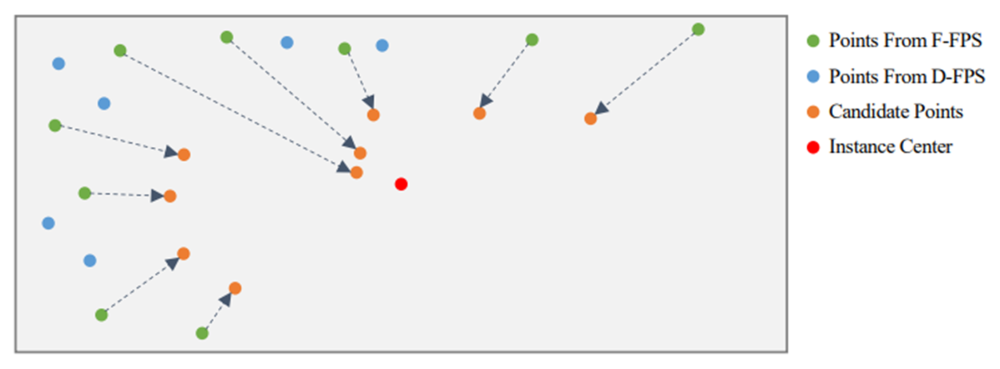
由于F-FPS，在SA层之后保留了不同实例中的大量正点。然而，由于总代表点数的限制，在下采样过程中丢弃了许多负点，这有利于回归但阻碍了分类。**也就是说，在SA层的分组阶段，从相邻点聚合特征，负点无法找到足够的周围点，从而无法扩大其感受野**。结果，该模型难以区分正负点，导致分类性能不佳。

如上所述，在SA层之后，不仅要尽可能多地采样正点，而且我们还需要收集足够多的负点以进行更可靠的分类。提出一种新的融合采样策略（FS），其中F-FPS和D-FPS在SA层期间应用，以保留更多用于定位的正点并保留足够的用于分类的负点。 具体来说，分别使用F-FPS和D-FPS采样并将这两个集合一起馈送到SA层中

#### 框预测网络

* 候选生成层

在由几个SA层实现的骨干网络与融合采样交织后，我们从F-FPS和D-FPS中获得了一个点子集，用于最终预测。在以前的基于点的方法中，应该在预测头之前应用另一个SA层来提取特征。一个普通的SA层包含三个步骤，包括中心点选择、周围点提取和语义特征生成。这里直接基于F-FPS生成的参考点来预测偏移量并将后续过程的参考改为偏移后的点。

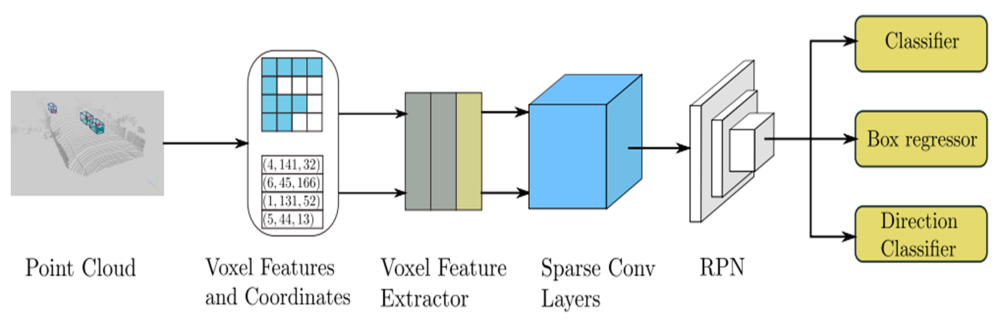


* **Anchor Free的box head**

对每个候选点直接预测对应实例的距离，大小和方向。

* **Anchor Free的box head + Centerness Loss**

### SECOND



* 体素化部分和VoxelNet基本上是一致的

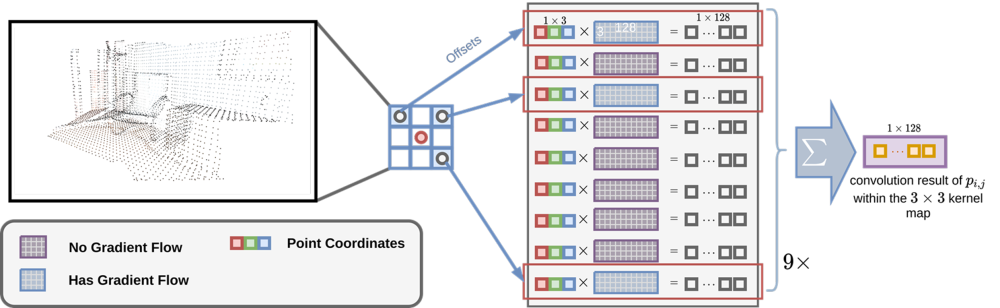
* 用**稀疏卷积来代替VoxelNet的Conv3D**,实现推理加速，现已成为**Voxel表示的标准方法**

* Box Head部分，做了一点微创新：

* + 方向回归，解决了0度和180度之间外观类似但loss很大的问题，使用**周期性的sin(θ)**来表示方位角，并对正反方向再进行而分类

* + 分类损失使用了Focal Loss

#### 稀疏卷积

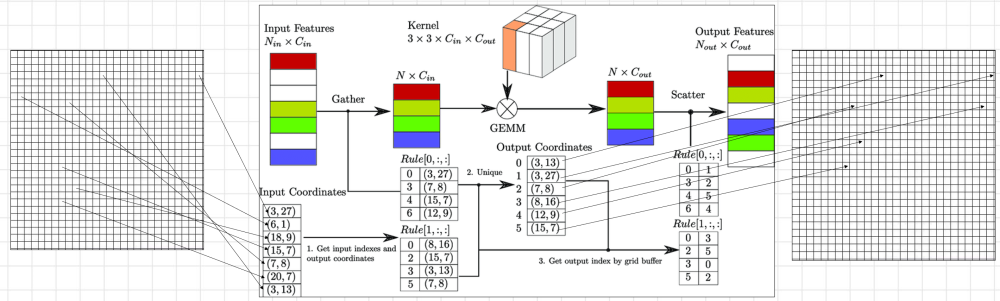


* 稀疏卷积的计算过程，是一个Gather-GEMM-Scatter的过程，计算稀疏卷积之前，将所有的需要计算的点的坐标收集起来，并将权重坐标也收集起来，拉伸为一个一维数组

* + Gather阶段，通过输入坐标拿到所有点的数值和权重的数值

* + GEMM阶段，计算输入与权重的乘积

* + Scatter阶段，通过输出坐标，将上一步的计算结果分散塞进输出的数组中



* Input Rule Book：就是从哪些坐标取出来输入，以及跟卷积核的哪个权重对应（im2col）

* Output Rule Book：表示将输出的运算结果塞到输入的哪个位置

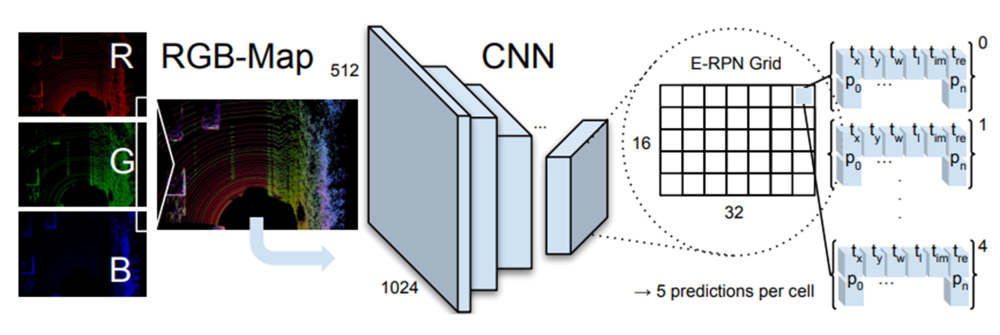
* 举例：

* + Input Rule：[0, (3, 27)] 卷积核的第0个权重对应输入特征的（3，27）位置（相乘）

* + Output Rule：[0, (3, 13)] 代表第0个计算结果，存入（3，13）这个位置

* + Input to Output Rule: [0, 1] 代表第0个计算结果对应第1个输出位置（输入->输出映射）

### YOLO3D/Complex-YOLO

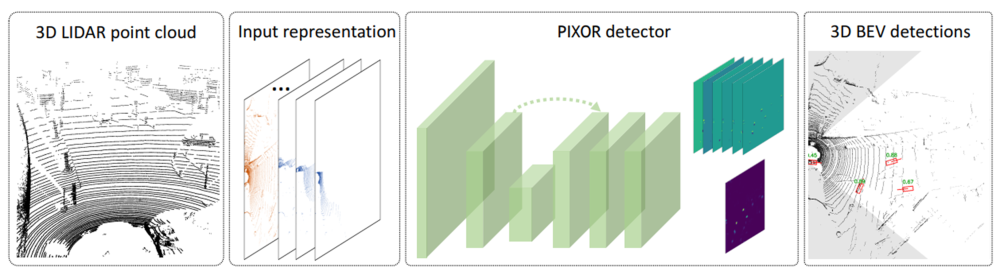


* 将点云转换为俯视图（**BEV**），大小为80m\*40m，分辨率为8cm，其中3个通道分别是**最大高度、最大强度、点云密度**，对应图像的RGB三个通道

* 使用YOLOv2作为backbone，以及6种anchor，并回归x, y, w, l, θ五个变量，其中θ同样使用的是类似**sin(θ)和cos(θ)**这样的方式

* 总体上来说与2D目标检测没有太大区别，**创新点就是投影方式**

### PIXOR



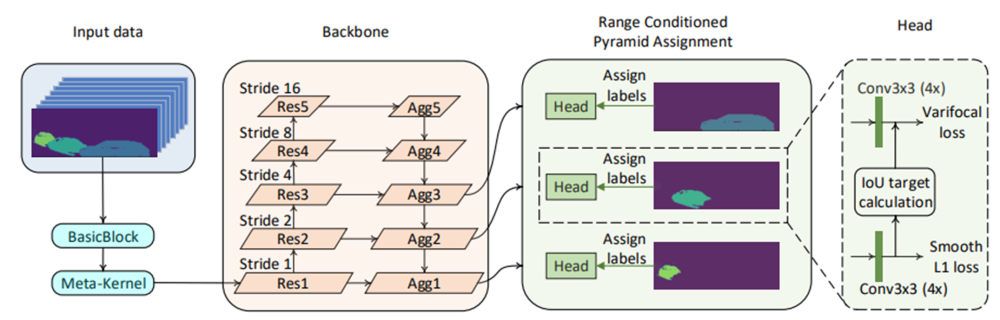
* 与complex-YOLO不同，PIXOR将点云投影到BEV视角，以占用率来表示（类似体素，将垂直方向格子看作通道）并增加3个通道，分别表示强度、超出范围的点，因此一共有35 + 3个通道

* + dL，dW和dH分辨是投影分辨率（0.1米），3代表强度、点云超出上边界、点云超出下边界



* Box Detection Head和Complex YOLO差不多，同样也是回归{𝑑𝑥, 𝑑𝑦,𝑤,ℎ,sin⁡(𝜃),cos⁡(𝜃) }以及分类，是一种anchor free的检测器

### RangeDet



* RangeView与BEV不同，它并不会出现点云重叠的问题

* 但如果使用RangeView接上2D CNN来做3D目标检测，效果并不好，与BEV视角相比AP约有10~20个点的差距

* RangeDet的输入：8个通道**(𝑖𝑛𝑡𝑒𝑛𝑠𝑖𝑡𝑦,𝑒𝑙𝑜𝑛𝑔𝑎𝑡𝑖𝑜𝑛,𝑥,𝑦,𝑧,𝑎𝑧𝑖𝑚𝑢𝑡ℎ,𝑖𝑛𝑐𝑙𝑖𝑛𝑎𝑡𝑖𝑜𝑛)**

* + 𝑖𝑛𝑡𝑒𝑛𝑠𝑖𝑡𝑦: 反射强度

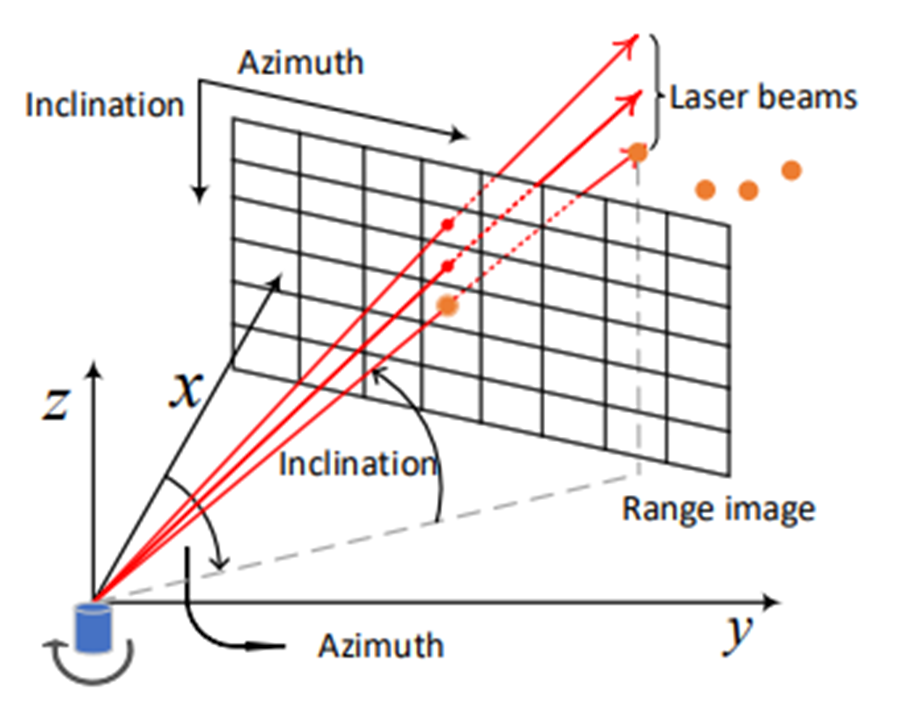
* + range：depth，深度，直线距离

* + elongation：延伸率，高延伸率低反射率的点一般是雾、灰尘、雨点

* + 为Waymo数据集所特有

* + - azimuth：水平方向的方位角，弧度

* + - inclination：垂直方向的方位角，弧度

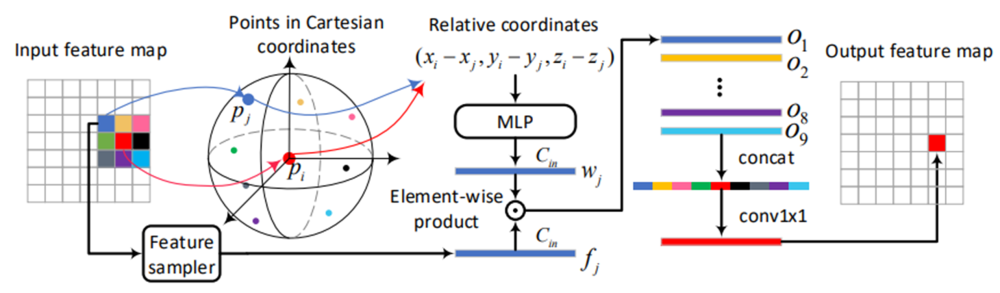


* RangeDet不是纯粹的将RangeView看作一个2D图像使用普通CNN来做目标检测，而是使用了Meta-Kernel卷积

* 使用了FPN多分辨率预测，并按照**物体距离而不是大小来决定放置在哪层FPN上**

* 后处理使用了WNMS（ 实际上就是重叠的多个框加权平均）以及数据增强，进一步提升效果

#### Meta-Kernel卷积



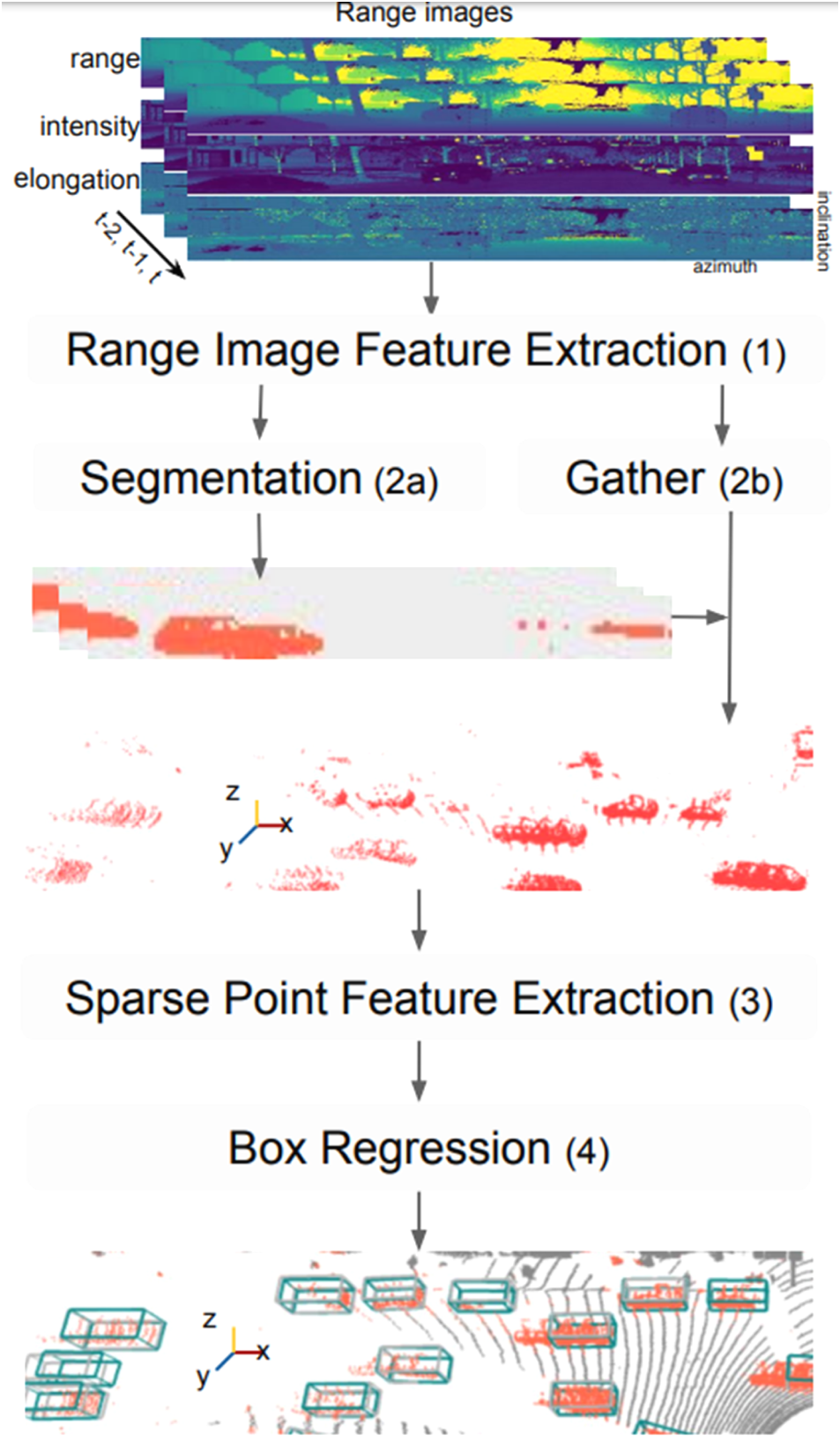
* Meta Kernel Convolution的计算方法

* 计算3x3邻域点与中心点的距离(Δx，Δy，Δz）并输入到MLP中，得到一组权重，并于相应的点相乘

* 得到9个点的特征后并不直接相加，而是concat到一起并再次经过一个3x3卷积融合到一起

结论：使用常规卷积做RangeView的目标检测是不可行的，必须在卷积核层面进行改造，才能达到跟BEV接近的精度

### Range Sparse Net



* Waymo在2021年发表的论文，性能比**pointpillars**更优

* RSN不是纯粹的RangeView检测方法，而是将RangeView和Voxel方法相结合

* RSN观察到在RangeView上做分割精度尚可，但做检测则效果不好，于是第一个阶段用一个类似Unet的结构对点云做分割，得到前景和背景，在第二个阶段用分割结果再去拿点云做检测。

* 在第二个阶段，将全部的前景点提取出来，并使用VoxelNet或者pointpillars做特征提取，再接上CenterPoint预测box

* 由于经过分割，前景点数大大减少，因此使用稀疏卷积可大幅提升运行速度，因此可以将检测范围做的很大（-79.5m~79.5m）。

* **总结：RSN实际上同时利用了RangeView和Voxel View，这种多View相结合的方式，是目前的一个发展趋势**。

### 单阶段总结

* Pointnet一般是作为体素网络的局部特征提取来使用的，很少作为完整的backbone使用

* **Pointnet++**可以作为完整backbone使用但运行效率极低，因此**在自动驾驶中很少使用**

* 基于体素的方式是较为常用的方法，其中**VoxelNet（SECOND）精度较好但耗时较长，而PointPillars精度略差但速度很快**

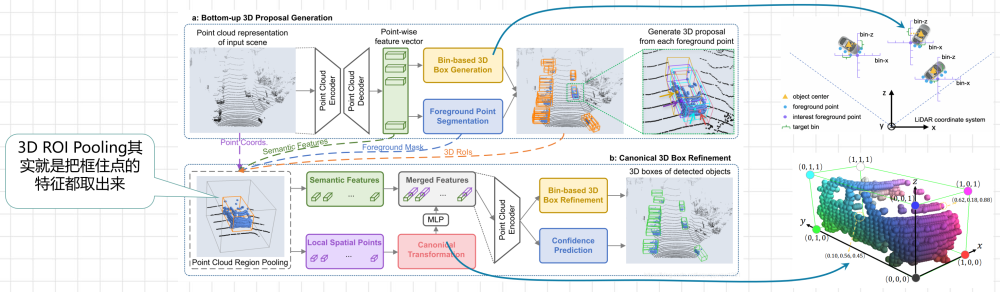
* 基于RangeView的投影方法，与基于体素方法相比，检测距离有优势

* 基于**BEV投影**的方法，不同方法的差异在于投影的设计方式，基本上可以被自动学习表示的pointpillars所代替

* 目前在自动驾驶中一般使用**基于Voxel的方法与基于RangeView的方法相结合，保证近距离的检测精度足够高，并且在较远距离处也有一定召回**

## 多阶段技术发展

### Point-RCNN



* backbone采用了pointnet++，也属于point based方法

* + 对提取特征后的每一个点，先预测是前景还是背景

* + 对于每个前景点都预测一个3D box，包括（Δx，Δy，Δz，h，w，l，θ）

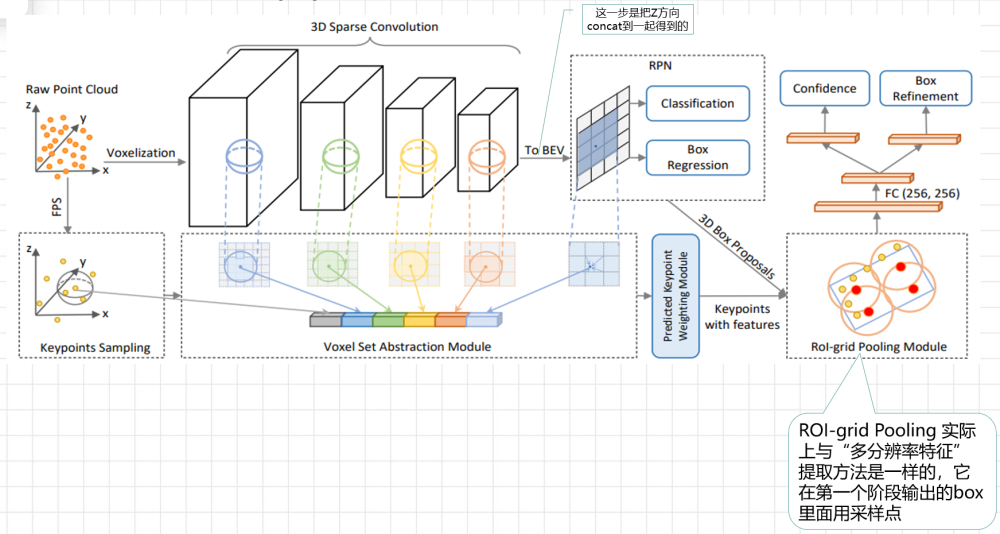
* + 对于Δx和Δy使用了bin-based方法，也就是类似anchor的预设偏移量

* 第一个阶段得到的框，再通过3D ROI Pooling来进行二阶段的refinement

* bin-based：就是将回归转换为分类问题，类似“anchor”的概念

* Canonical Transformation:就是将框“正过来”，统一调整到一个固定朝向

### PV-RCNN



* 同时结合了Point表示和Voxel的表示

* 第一个阶段，与**SECOND网络**是一致的，体素化+3DCNN直接输出box

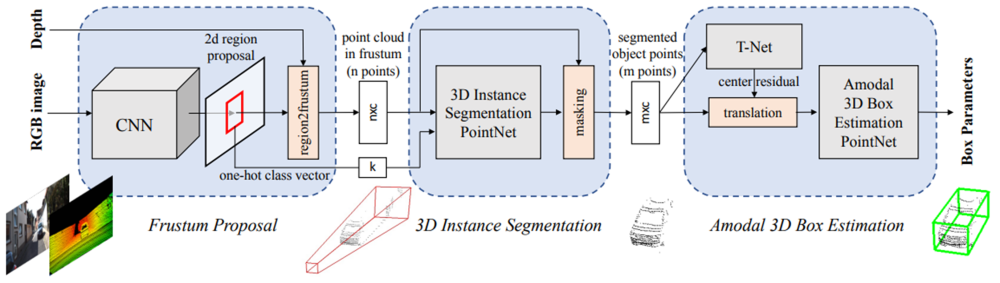
* 第二个阶段融合了三种特征：**原始点云、多分辨率特征、BEV特征**

* 多分辨率特征：使用FPS得到采样点，再把采样点周围的体素特征拿出来

* + BEV特征：把采样点周围的像素特征拿出来

* + 三种特征concat到一起，作为第二阶段ROI Grid Pooling的输入

### Frustum-PointNet



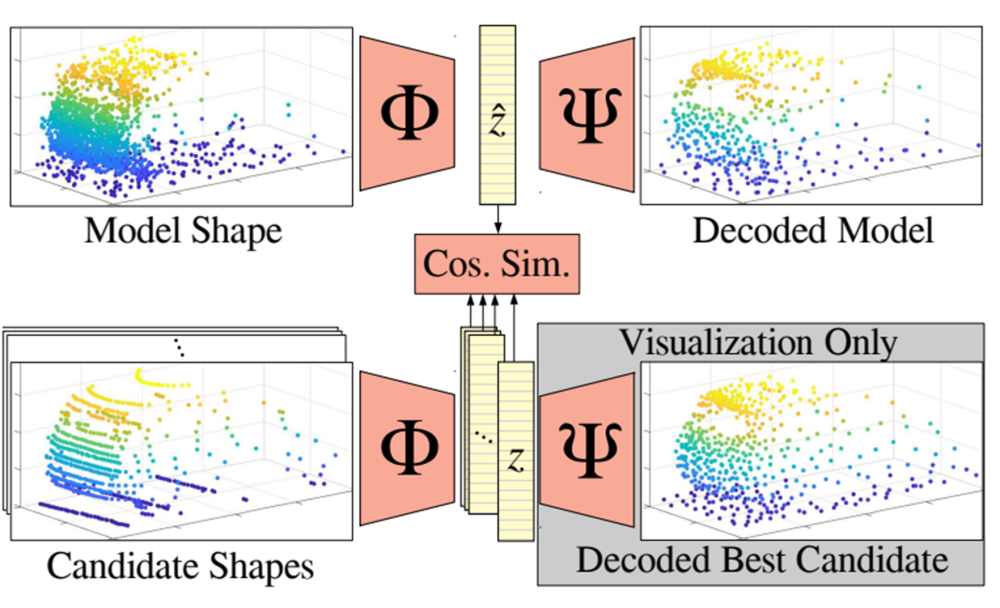
* 先利用SSD在图像上预测2D框，并将框按照视锥投影到点云上，将视锥内的点云全都取出来，并经过一个pointnet分割网络去掉背景点

* 将剩余的前景点经过一个T-Net，转换到一个“标准视角”

* Amodal 3D Box PointNet：预测物体的(∆𝑥,∆𝑦,∆𝑧,ℎ,𝑤,𝑙,𝜃)以及分类

## 3D目标跟踪

### 3D Siamese Tracking



* 借鉴了2D目标跟踪的SiameseFC的思想，通过孪生网络来对目标和候选区域的点云提取特征，并计算相似度

* 与2D目标跟踪的相关卷积类似，3D Siamese计算的是特征的**余弦相似度**（相关卷积可以理解为2D的余弦相似度）

* 与2D目标跟踪不同的是，**它需要先有3D检测结果，而不能通过先前帧的结果在新的点云上直接输出3D框，相当于只是做匹配**

## 总结

* 激光雷达感知中，**3D目标检测**是L4自动驾驶的必要模型，也是最重要的模型

* 除了3D目标检测外，其他信号均可以通过高精地图定位、V2X等方式获取，在L4级别自动驾驶中并不一定是必须的

* 由于图像2D目标检测非常成熟，因此点云3D目标检测倾向于将它转化为一个2D目标检测问题来做

* **BEV的优势是精度高，劣势是检测距离有限；Range View的精度不高但检测距离没有限制。两者相结合可以得到比较好的结果**

* 目前点云与图像融合的方式，与单独使用点云来做检测，并没有明显优势，甚至还会损害精度

* 自动驾驶3D感知，**PointPillars/CenterPoint** 是目前较为流行的方案