Color of Pointcloud Matters!

3D Object Detection by Colorful Pointcloud Alignment Between Lidar Pointcloud and RGB Image

Saeed Firouzi
Sharif University of Technology
saeedmr881@gmail.com

Abstract

One of the most critical task in autonomous vehicle is to detect different kind of object in the environment. Detecting these objects are important due to different reaction we have with them. Finding these objects without the colors or fusing lidar pointcloud with image , without aligning color of pixels of image to the lidar pointcloud, is hard for the model to understand it. But in this work, we first align image and pointcloud lidar and then give a color to the every poitcloud, and then with these colorful pointcloud we can detect different objects with a voxel-based CNN-Transformer model that has a better performance to the other models.
The training code and pre-training model are available at
https://github.com/saeed5959/3d-object-detection
to help open source community.

1 Introduction

One of the most critical task in autonomous vehicle is to detect different kind of object in the environment. We can divide these objects into 2 categories: 1-active objects 2-inactive objects. Objects like rock, box, wall, tree, … are all inactive objects that can not move and the only important thing about them is to not collide with them. With lidar pointcloud you can detect these and avoid them. But car, cyclist, human, traffic line, … are active objects that some of them can move and some of them have interaction with environment. Detecting these objects are important due to different reaction we have with them.

If we detect a car in the path that is moving, we can be in behind of that car or we can switch the line. If we detect a human in the middle of the road we must reduce our speed and change the line to avoid the collision. Or if we detect a traffic line we should find the color of it; red for stop and green for movement.

So beside of avoiding the collision, there are some other task when a car encounters with active objects. Here, detecting these active objects make this task more important.

Old methods, use just the lidar sensor to get a accurate 3d pointcloud from the environment, but these pointcloud are without color that make it very hard for model to detect objects. Fortunately, new method “fusion” use a camera beside of lidar sensor to

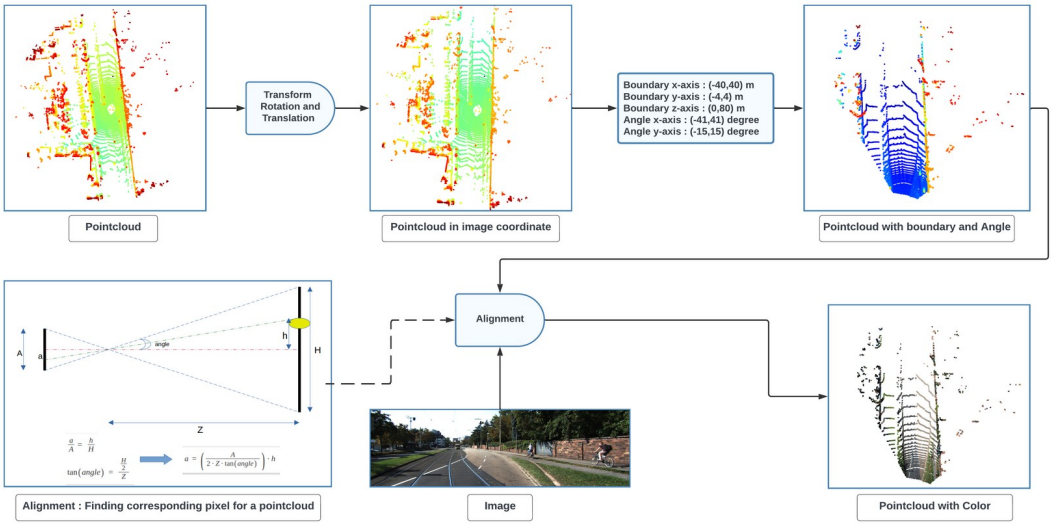

Figure 1 : **Alignment between pointcloud and image**. Giving the color of pixel in image to the corresponding pointcloud

take a picture from environment. They align this camera with lidar sensor to use both data in their model but separately. This method gives both lidar pointcloud and image to the model and model tries to make a connection between pixels of image and pointcloud. This indirect connection affect the performance of the model. But in our method we take away this barrier and directly make a connection between them.

The critical thing in here is that this alignment between camera and lidar sensor must be accurate.

Our main contributions is:

- Making a colorful pointcloud from alignment between image and lidar sensor
- Propose a new method in 2d backbone

2 Alignment

Lidar sensor take data from the environment in 360 degree. But camera can only take image from a part of environment and in a specific angle. For making this alignment first of all, we should transform lidar pointcloud from lidar coordinate to the camera coordinate. And then we should find the boundary and then find the angle of image in y-axis and x-axis. And then for any pointcloud, we should calculate relative position in y-x plane based on z-axis data (depth) and transform it to the image plane and assign the color of corresponding pixel to that pointcloud.
The overall process of alignment can be seen in Figure 1.

3 Model Architecture

3.1 Voxelization

Due to sparse density and being empty in most of area in the 3d environment, for reducing process calculation we have to make a voxelization and gather some points in small voxel.

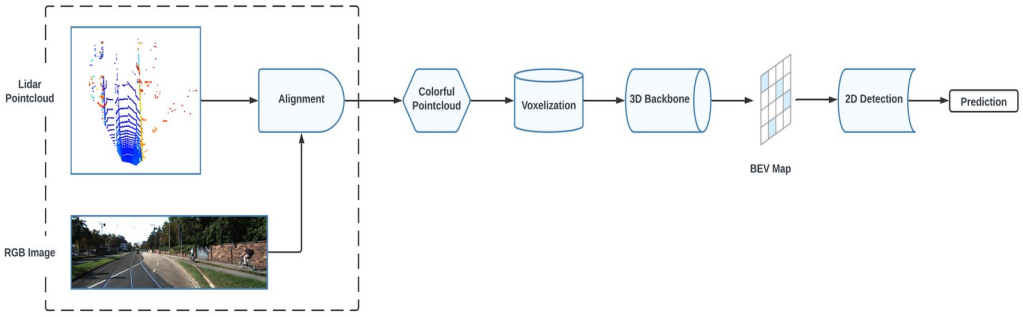

Figure 2. **An illustration of model architecture**

The size of this voxel must be in a trade off between process time and accuracy and size of other objects. This voxel shouldn't be bigger than any other objects. Our Range of lidar sensor is about x=(-40,40) m, y=(-4,4) m, z=(0,80)m, and the our objects are human, car, cyclist, barrier and traffic line. A good choice for size of voxel is [5cm, 10cm, 5 cm].
By dividing all environment in this boundary to the voxels with above size and put any points in their corresponding voxel we would have a voxelize pointcloud.
In any voxel there is 5 parameters as their features: mean of color(red, green, blue), intensity, mean of y-axis. Because we want to convert 3D voxel to A BEV map that cause to compress y-axis, then using mean of y-axis in the parameters can be useful.

3.2 3D Backbone

For 3d backbone we have used the Resnet[1] structure and instead of 2D conv filter we have used 3D conv filter. The padding would be the same and there would be 3 block followed by 3 maxpooling to downsample our environment 8x times. Then approximately the size of voxel would be [40cm, 80 cm, 40cm] which this size is sufficient for detecting the desired objects.
After 3D backbone block we will concatenate all voxel in y-axis to convert it to a BEV map[2], Because the size of object in y-axis is less important than to z-axis and x-axis.
The 3D backbone structure can be seen in Figure 3.

3.3 2D Detection

Our 2D backbone is similar to [3]. First of all, we use a CNN network with 2D conv filter with size of 5 to increase receptive fields. Then we prepare these voxel vectors for transformer block by putting in a raster order and adding positional embedding. Then with concatenating output of transformer for voxel vectors with POA matrix we can predict objectness and class and bounding box of a object in any voxel.
The 2D backbone structure can be seen in Figure 4.

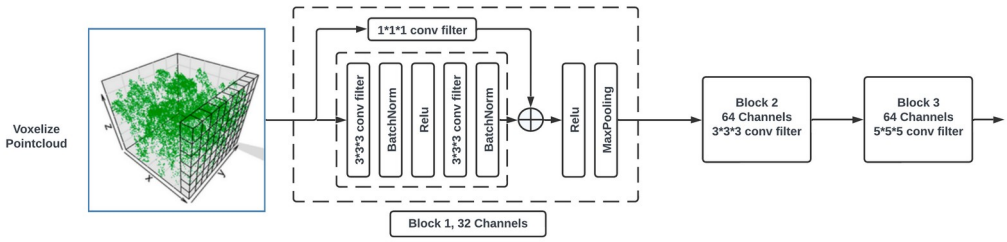

Figure 3. **An illustration of 3D backbone**

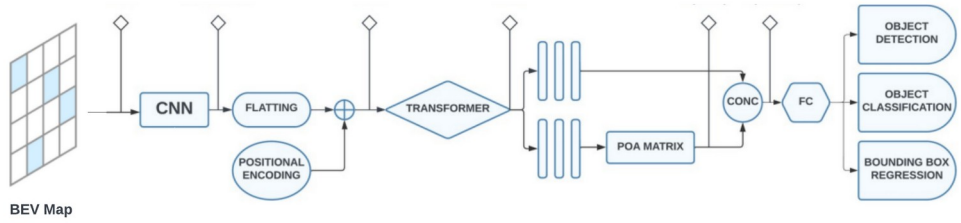

Figure 4. **An illustration of 2D backbone**

4 Training

Based on the new strategy that was introduced in [3], we use segmentation dataset alongside of detection dataset to choose voxels that have intersection with object, not just the center voxel of object or all voxels of objects in a 3d bounding box.
This approach helps to ignore misinformation that may come from voxels inside the box that does not contain the object.

5 Experiment

We train the model on KITTY dataset [7] that includes 7128 images with 5 categories. Every data contains a bounding box for objects and a segmentation data. We evaluate our model based on two factors; accuracy and inference time. We use a batch size 32 and train on V-100 gpu.

Our Model	Precision-0.2	Recall-0.2	Inference time (fps)
X epoch			

The result will be added soon!!!

6 Conclusion

we introduced our model as a unified and one-stage model that can use image as a color data to align with pointcloud data and make a colorful pointcloud. This approach will help to identify object better and more accurate.
Our model is fast and can detect objects in different shape and size accurately.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition, 10 Dec 2015

[2] Jorge Beltrán, Carlos Guindel, Francisco Miguel Moreno, Daniel Cruzado, Fernando García, Member, IEEE and Arturo de la Escalera, BirdNet: a 3D Object Detection Framework from LiDAR information, May 2018

[3] Saeed Firouzi from Sharif University of Technology, Place of Attention Matters! An End-to-End Object Detection with Vision Transformation, Jun 2023, https://github.com/saeed5959/object-detection-transformer