3-D Urban Objects Detection and Classification From Point Clouds

Yassin Alkhalili, Manisha Luthra, Amr Rizk, Boris Koldehofe
KOM - Multimedia Communications Lab
firstname.lastname@kom.tu-darmstadt.de
About me

- Phd student at KOM - Multimedia Communications Lab – Tu-Darmstadt
- I am part of “MAKI– Multi-Mechanisms Adaptation for the Future Internet”
- Research Interest: Point cloud Processing and Streaming

Immersive Communications

Image source: https://youtu.be/7d59O6cfM0
How to define a Point Cloud?

- A collection of points in 3D space

---

One point: X, Y, Z, R,G,B

Point cloud format (Example)

---

Image source: https://es.mathworks.com/help/vision/ref/pfwrite.html
Motivation

- Point Cloud can be created by LiDAR
  - Easy to produce

- LiDAR VS RADAR and IR
  - insensitivity to colors, lighting conditions, and radial distortion

Row Data

Knowledge about the environment

Image source: https://velodynelidar.com/hdl-32e.html
https://ori.ox.ac.uk/efficient-object-detection-from-3d-point-clouds/
This work addresses the problem of 3D Urban object class recognition in different LiDAR scenes.
Challenge Dataset

Bird’s eye view of a LiDAR scene
Challenge Dataset

Single Object Scene

Multiple Objects Scene
Challenge Dataset

- The dataset includes 28 different object classes
- 50 single object scene per object class → 1400 single object scenes
- 600 multiple object scenes
- Each multiple object scenes contains 10 to 50 objects

- Labels are assigned to the entire scene and not the individual objects.

[ATM, Car, Pedestrian]
Challenges in the Dataset

- High number of objects per scene (10-50)

- Objects are in different distance to the LiDAR → different object scales and different point densities

- Possibly interwoven objects

- Object classes are very similar to each other

- The amount of training data 600 multiple objects scenes are limited
Goal: learn object types and recognize them in low latency

Proposed method:

1. Detecting candidate objects in a given scene
2. Generating features for each object
3. Use classification model for training and predicting
1- Detecting Candidate Objects in a Given Scene

Given Scene

Removing walls and floors

$X$ and $Z \in [-25, 25]$

$Y > -2.5$

Clean Scene – 60% data less

Euclidean Segmentation

```java
clustering.setClusterTolerance(0.3); // 30 cm
clustering.setMinClusterSize(50);
clustering.setMaxClusterSize(5000);
```
2- Object Features Generation
Ensemble of Shape Functions (ESF) [1]

**ESF** shape descriptor based on three distinct shape functions describing:
- Point Distance distribution
- Angle distributions
- Area distribution

ESF Descriptor

Why ESF [1]:

- Scale to hundreds of object types
- No preprocessing necessary e.g. surface normal calculation
- Handles data error such as outliers holes noise gracefully

2- Object Features Generation

![Diagram showing object features generation with ESF and BMWX5Simple data]

- 0.662
- 1.987
- 1.987
- 1.325
- 1.325
- 1.325
- 1.325
- 1.987
- 3.974
- 1.325
System Model

Point Cloud Scene → Point Cloud Processing Unit → Object Feature Generator → Training or predicting
Training vs Predicting

Training

Predicting
Evaluation

- We used single object scenes (labelled data) to train the model

  - Training Data:
    - 767 objects
    - 28 object types
  - Testing Data:
    - 330 objects

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>K Neighbors</td>
<td>0.48</td>
<td>0.50</td>
<td>0.49</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.59</td>
<td>0.61</td>
<td>0.60</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>0.49</td>
<td>0.52</td>
<td>0.50</td>
</tr>
</tbody>
</table>
## Evaluation

<table>
<thead>
<tr>
<th>Class</th>
<th>Precision</th>
<th>Recall</th>
<th>Avg. points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atm</td>
<td>0.67</td>
<td>0.60</td>
<td>245</td>
</tr>
<tr>
<td>Bench</td>
<td>0.47</td>
<td>0.69</td>
<td>172</td>
</tr>
<tr>
<td>BigSassafras</td>
<td>0.85</td>
<td>0.79</td>
<td>970</td>
</tr>
<tr>
<td>BmwX5Simple</td>
<td>0.42</td>
<td>0.62</td>
<td>874</td>
</tr>
<tr>
<td>ClothRecyclingContainer</td>
<td>0.38</td>
<td>0.40</td>
<td>437</td>
</tr>
<tr>
<td>Cypress</td>
<td>0.90</td>
<td>1.00</td>
<td>598</td>
</tr>
<tr>
<td>DrinkingFountain</td>
<td>0.00</td>
<td>0.00</td>
<td>53</td>
</tr>
<tr>
<td>ElectricalCabinet</td>
<td>0.45</td>
<td>0.50</td>
<td>101</td>
</tr>
<tr>
<td>EmergencyPhone</td>
<td>0.88</td>
<td>0.94</td>
<td>131</td>
</tr>
<tr>
<td>FireHydrant</td>
<td>0.00</td>
<td>0.00</td>
<td>48</td>
</tr>
<tr>
<td>GlassRecyclingContainer</td>
<td>0.88</td>
<td>0.88</td>
<td>417</td>
</tr>
<tr>
<td>IceFreezerContainer</td>
<td>0.36</td>
<td>0.45</td>
<td>270</td>
</tr>
<tr>
<td>Mailbox</td>
<td>0.22</td>
<td>0.29</td>
<td>90</td>
</tr>
<tr>
<td>MetallicTrash</td>
<td>1.00</td>
<td>0.25</td>
<td>69</td>
</tr>
<tr>
<td>MotorbikeSimple</td>
<td>0.54</td>
<td>0.64</td>
<td>136</td>
</tr>
<tr>
<td>Oak</td>
<td>0.78</td>
<td>0.78</td>
<td>410</td>
</tr>
<tr>
<td>OldBench</td>
<td>0.50</td>
<td>0.11</td>
<td>108</td>
</tr>
<tr>
<td>Pedestrian</td>
<td>0.50</td>
<td>0.43</td>
<td>112</td>
</tr>
<tr>
<td>PhoneBooth</td>
<td>0.79</td>
<td>0.69</td>
<td>467</td>
</tr>
<tr>
<td>PublicBin</td>
<td>0.47</td>
<td>0.80</td>
<td>81</td>
</tr>
<tr>
<td>Sassafras</td>
<td>0.50</td>
<td>0.56</td>
<td>640</td>
</tr>
<tr>
<td>ScooterSimple</td>
<td>0.67</td>
<td>0.91</td>
<td>150</td>
</tr>
<tr>
<td>ToyotaPriusSimple</td>
<td>0.73</td>
<td>0.69</td>
<td>728</td>
</tr>
<tr>
<td>Tractor</td>
<td>0.60</td>
<td>0.40</td>
<td>456</td>
</tr>
<tr>
<td>TrashBin</td>
<td>0.43</td>
<td>0.21</td>
<td>114</td>
</tr>
<tr>
<td>TrashContainer</td>
<td>0.56</td>
<td>0.33</td>
<td>338</td>
</tr>
<tr>
<td>UndergroundContainer</td>
<td>0.25</td>
<td>0.44</td>
<td>97</td>
</tr>
<tr>
<td>WorkTrashContainer</td>
<td>0.69</td>
<td>0.50</td>
<td>306</td>
</tr>
</tbody>
</table>
Evaluation

- Performance of the our system as reported by DEBS’19 online evaluation system

<table>
<thead>
<tr>
<th>System</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>Runtime (sec)</th>
<th>#Scenes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOC_KOM</td>
<td>0.479</td>
<td>0.56</td>
<td>0.67</td>
<td>557</td>
<td>442</td>
</tr>
</tbody>
</table>
System Limitations

- **Segmentation**
  - might divide the same objects into many
  - Very closed objects to each other might be considered one object

- **Execution time**
  - We develop our solution in Python, but ESF is available only in C++, so we had to call C++ from python which impose a delay which can be easily avoided if we implement the whole solution in C++
Questions

Yassín Alkhalili, M.Sc.

Yassín.Alkhalili@KOM.tu-darmstadt.de
Rundeturmstr. 10
64283 Darmstadt/Germany
www.kom.tu-darmstadt.de

Phone +49 6151 16-20491
Fax +49 6151 16-29109
Viewpoint Feature Histogram (VFH) [1]
Summary
Future Work