XODRParser: A Python tool for working with OpenDRIVE formats

Intern: Li Liu
Supervisor: Rainer Kauschke

HELLA GmbH Co. KGaA
Lippstadt, Germany

08th November, 2022

Figure 1. Visualization of entire road network.

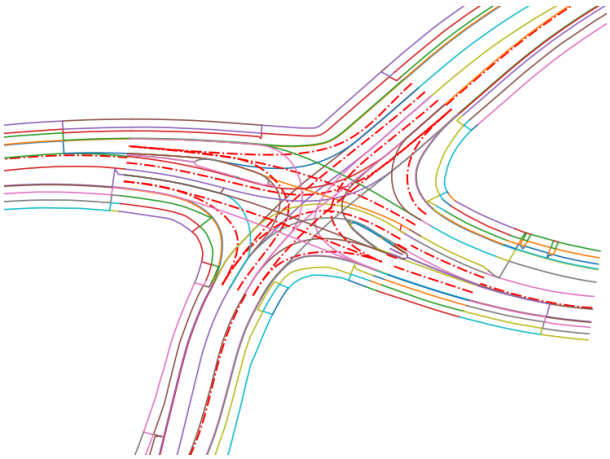

Figure 2. Details of a road intersection.

I Introduction

The ASAM OpenDRIVE format provides a common base for describing road networks with extensible markup language (XML) syntax, using the file extension xodr. The data that is stored in an ASAM OpenDRIVE file describes the geometry of roads, lanes and objects, such as roadmarks on the road, as well as features along the roads, like signals. The road networks that are described in the ASAM OpenDRIVE file can either be synthetic or based on real data. Road data may be manually created from road network editors, conversion of map data, or originate from converted scans of real-world roads.

During the internship, the author needs to conduct pre-development of automatic driving functions based on data in OpenDRIVE format. There are currently a few open source toolkits for this format, but they are either based on the C++ programming language, which is more difficult to test and use than python, or their functions are not perfect to match the concrete needs, making it not preferred to be used.

II Main functions developed

1 Visualization of the entire OpenDRIVE file

An OpenDRIVE file usually contains the geometric description of multiple Raod segments. The entire file can be easily visualized using the visualize_file() function. Processing time is between 5s to 30s depending on file size. A sample result is shown in Figure 1. The details of a road intersection are shown in Figure 2.

```
<road name="" length="3.758390062546e+01" id="1006000" junction="-1">
  <link>
    <predecessor elementType="junction" elementId="3154000" />
    <successor elementType="junction" elementId="3130000" />
  </link>
```

Figure 3. "link" tag of OpenDRIVE file.

2 Find a road segment by ID

Using XODRParser can easily find the corresponding road segment according to the ID value. This function retruns a object of Class "Road Parser", which can be use to compute further lane coordinates of this road segment.

3 Find the predecessor and successor

User can easily find the predecessor and successor defined in OpenDRIVE file of a certain road segment given it's ID. This information comes from the "link" tag of OpenDRIVE file. An example XML code is shown in Figure 3.

4 Process the given GPS coordinates

Find a road segment that contains GPS coordinates, and verify that the coordinates are in the left or right lane, as shown in Figure 4. If necessary, obtain the geometric information of the road reference line within the next specific distance (distance can be customized) corresponding to the GPS coordinates, as shown in Figure 5.

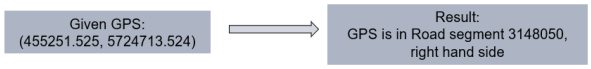

Figure 4. Find the road segment that contains a certain GPS coordinate.

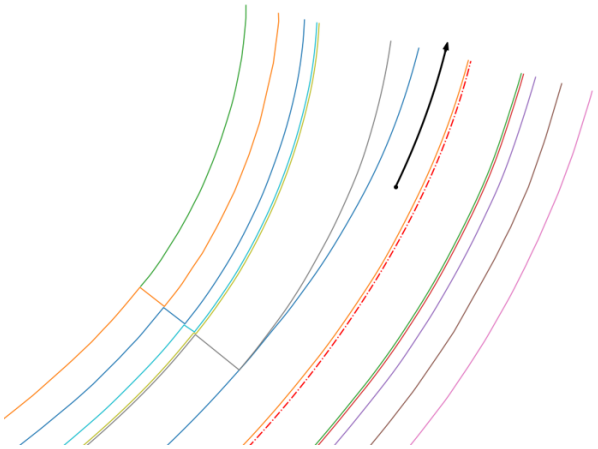

Figure 5. Find reference line geometry in a certain distance.

5 Collect the geometry parameters and calculate the coordinates of road lanes

As we all know, in OpenDRIVE only the geometry parameters of lanes are defined. So it's natural to first parse all these parameters and save them. Then calculate the concrete coordinates according to the geometry relations between these elements. With XODRParser the procedures we have just mentioned can be easily achieved using parse_lanes_parameters() and calculate_lanes_coordinate() of object "Road Parser".

6 Collect the geometry parameters and calculate the coordinates of road objects

Similar as last subsection, the objects coordinates can also be easily accessed. But it should be mentioned that there are many kinds of objects in OpenDRIVE, including angular object, circular object, and more details can also be attached to them, like repeating objects and object outline. Since this part is not very important for the author's internship task, only the center coordinates of the objects are calculated here, and other attributes, such as radius, height, length, etc., are ignored. Example result is shown in Figure 6, all types of objects appear indiscriminately in blue.

III Some examples of use

Some main function usages are listed here. The usage of other functions is relatively simple, which is omitted here.

1 Visualization of the entire OpenDRIVE file

```
from xodr_parser import XODRParser

path = "example.xodr"
parser = XODRParser(path)
parser.visualize_file()
```

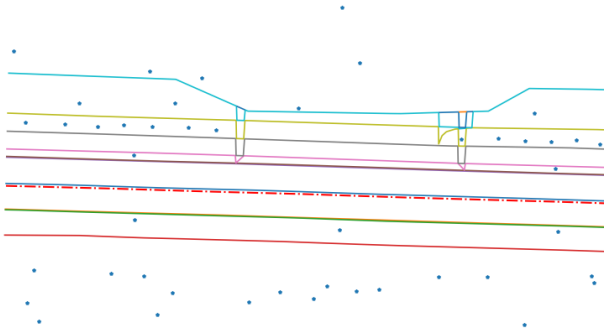

Figure 6. Objects on the road.

2 Find a road segment by ID

```
from xodr_parser import XODRParser

path = "example.xodr"
parser = XODRParser(path)
road_parser = parser.find_id('1002000')
road_parser.parse_lanes_parameters()
road_parser.calculate_lanes_coordinates()
print(road_parser.lanes_coordinates)
```

3 Find the road segment that contains the GPS coordinate

```
from xodr_parser import XODRParser

path = "example.xodr"
parser = XODRParser(path)
gps = [455262.18, 5724680.20]
exist, id, lane, road_parser =
    parser.find_road_from_gps(gps)
```

IV An easily overlooked detail

When user tried to get the road segment that contains the given GPS coordinate with find_road_from_gps(gps), he maybe gets more than one road segments, because some road segments, especially in junction regions, will overlap with each other. By default, the user will only get the road segment found at first. For more detailed information please refer to the official OpenDRIVE documentation.`https://www.asam.net/standards/detail/opendrive/.`

References

https://www.asam.net/
http://www.opendrive.org/
https://odrviewer.io/
https://www.3d-mapping.de/home/
https://lxml.de/tutorial.html/
https://stackoverflow.com/

APPENDIX A

1 Details of class XODRParser

Attributes	Details
file_path	The file path of XODR file
tree	Tree obtained with xml.etree.ElementTree
root	Root of the xml.etree.ElementTree
roads	Dict with ID as key and RoadParser as value
Functions	**Details**
find_id	Return a object of RoadParser given the ID
find_road_from_gps	Return a RoadParser object where the given GPS coordiante belongs and indicate the driving direction
calculate_projection_geometry	Return the geometric information of the road reference line within the next specific distance begin from the GPS coordinates
visualize_file	2D plot of road segments contained in the XODR file
find_predecessor	Return the ID of the predecessor given a road ID
find_successor	Return the ID of the successor given a road ID

2 Details of class Roadarser

Attributes	Details
road_node	Road node in XODR file obtained with xml.etree.ElementTree
geometry_nodes	Geometry node of this road segment in XODR file obtained with xml.etree.ElementTree
id	ID of the road segment
road	Road object of this road segment with alle sub-attributes saved in it
already_gathered	Boolen that indicates if road parameter is already parsed
road_length	Length of this road segment
s_coordinates	An array representing the road reference line
ref_line_coordinates	A list contains the coordiantes of road reference line
center_line$_coordinates$	A list contains the coordiantes of road center line
lanes_coordinates	Dict with lane id as key and coordinates of the corresponding lane as value
lane_offset_s	A list contains the starting s value of each lane offset
lane_section$_s$	A list contains the starting s value of each lane section
geometry_s	A list contains the starting s value of each geometry element in planView
elevation_s	A list contains the starting s value of each elevation element of road reference line
max_lane_id	Max lane id
min_lane_id	Min lane id
obj_coordinates	A list contains the center coordinates of objects on the road segment
predecessor	ID of the predecessor of this road segment
successor	ID of the successor of this road segment
Functions	**Details**
get_road_length	Return the length of this road segment
parse_lanes_parameters	Parse the parameters of lanes and save them in a object of class Road
parse_object_parameters	Parse the parameters of objects and save them in a object of class Road
calculate_lanes_coordinate	Calculate and save the coordinates of lanes
calculate_objects_coordinate	Calculate and save the coordinates of objects
plot_lanes_2d	2D plot of this road segment

3 Details of other classes

The definitions of other classes are almost the same as the official documents, including PlanView, Road, Lanes, etc. Please refer to the official documents on OpenDRIVE website.