

Lanelet2 Maps

Communtech/AVIN Project Summary

Michał Antkiewicz

Waterloo Intelligent Systems Engineering Lab

University of Waterloo

2022-03-08



Phase 1: Map generation

Goal: "Develop semi-automated procedure to translate Ecopia HD Map into Lanelet2 format"

- Developed an iterative three-step approach
 1. Map design using QGIS and tagging using "lanelet2_utils" script
 - Tagging stores Lanelet2 structure using JSON annotations and incrementally modifies the resulting Lanelet2 map, which allows for testing
 2. Map preparation using QGIS toolbox and our custom algorithms
 - Densifying features (adding points at a set minimum interval, e.g., 10 m)
 - Setting elevation from raster or other sources
 - Creates temporary layers
 3. Map generation using "lanelet2_utils" script
 - Recreates Lanelet2 map from scratch using the JSON annotations as a guide

1. Map design using QGIS



1. Map design using QGIS

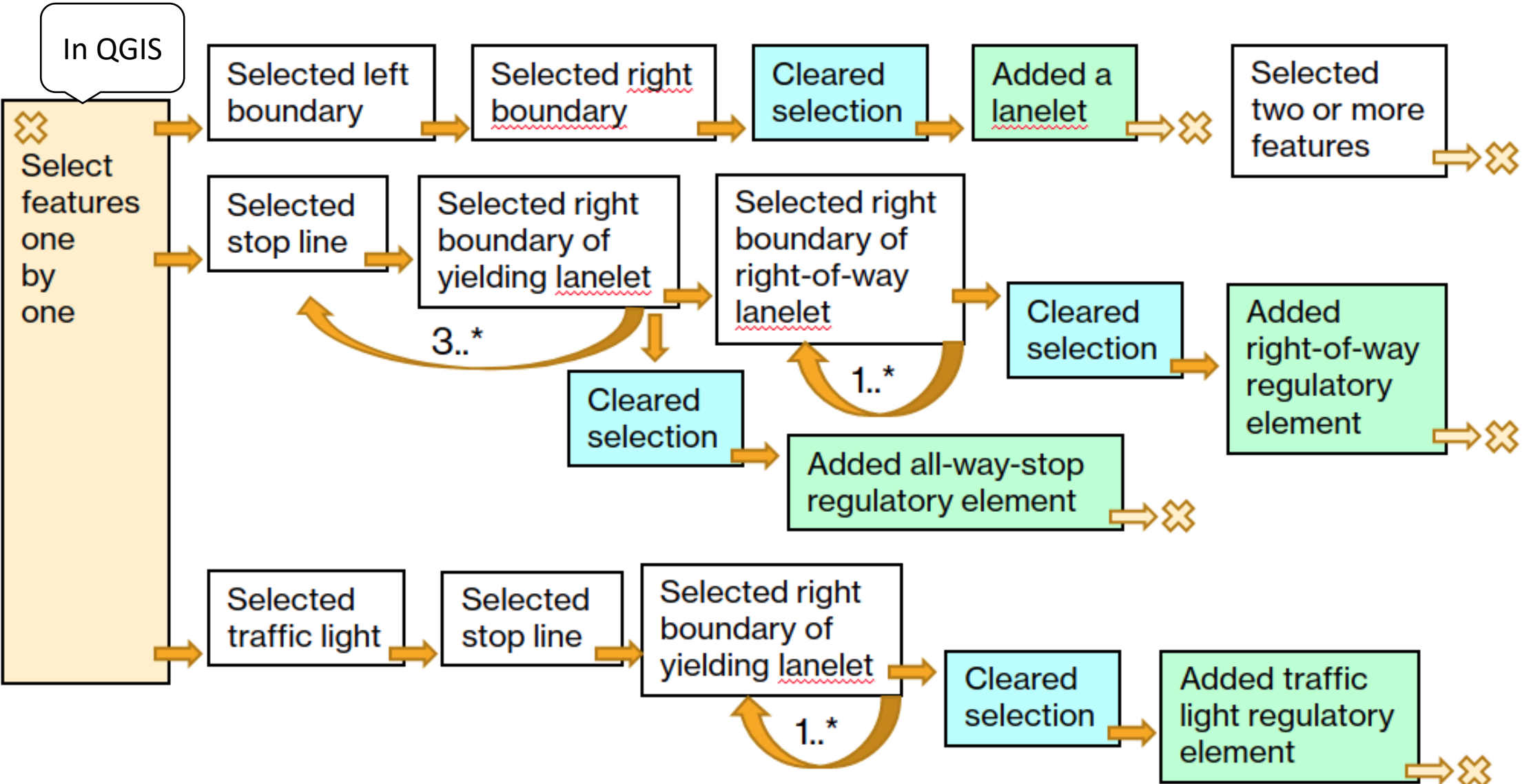


1. Map design using QGIS

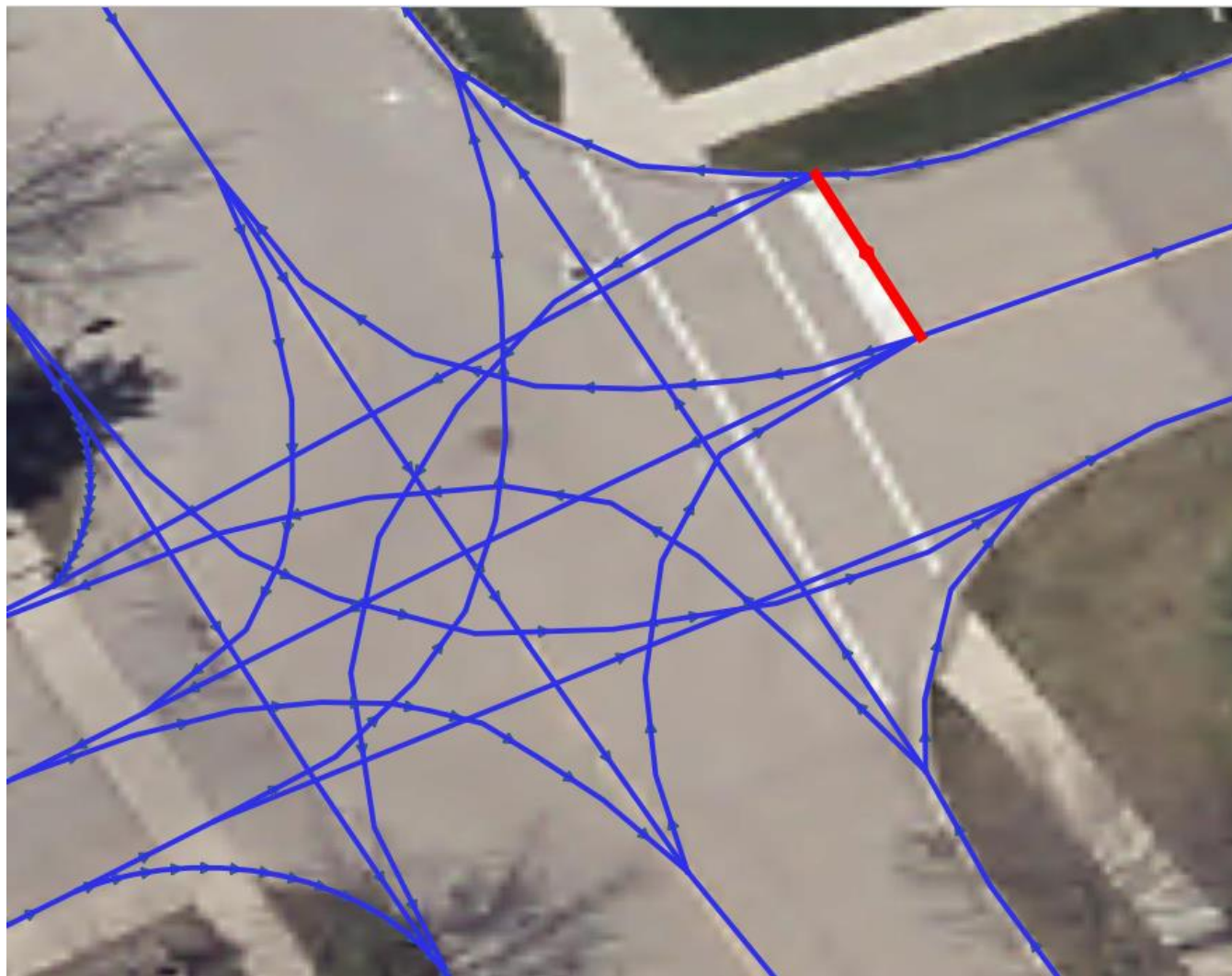


Connected parts of Westmount,
Columbia, Ring Road

1. Tagging using "lanelet2_utils" script



1. Tagging using "lanelet2_utils" script



Identify Results

Feature	Value
▼ Waterloo_Region	
catetitle	stop_line
▶ (Derived)	
▶ (Actions)	
fid	316455
catetitle	stop_line
rel	{"yielding": [323534], "right_of_way": [323550, 323553, 323584, 323557, 323576]}

JSON tags encode Lanelet2 structure

1. Tagging using "lanelet2_utils" script



Feature	Value
Waterloo_Region	
catetitle	tr
(Derived)	
(Actions)	
fid	324169
catetitle	traffic_light
rel	{"stop_line": 323500}

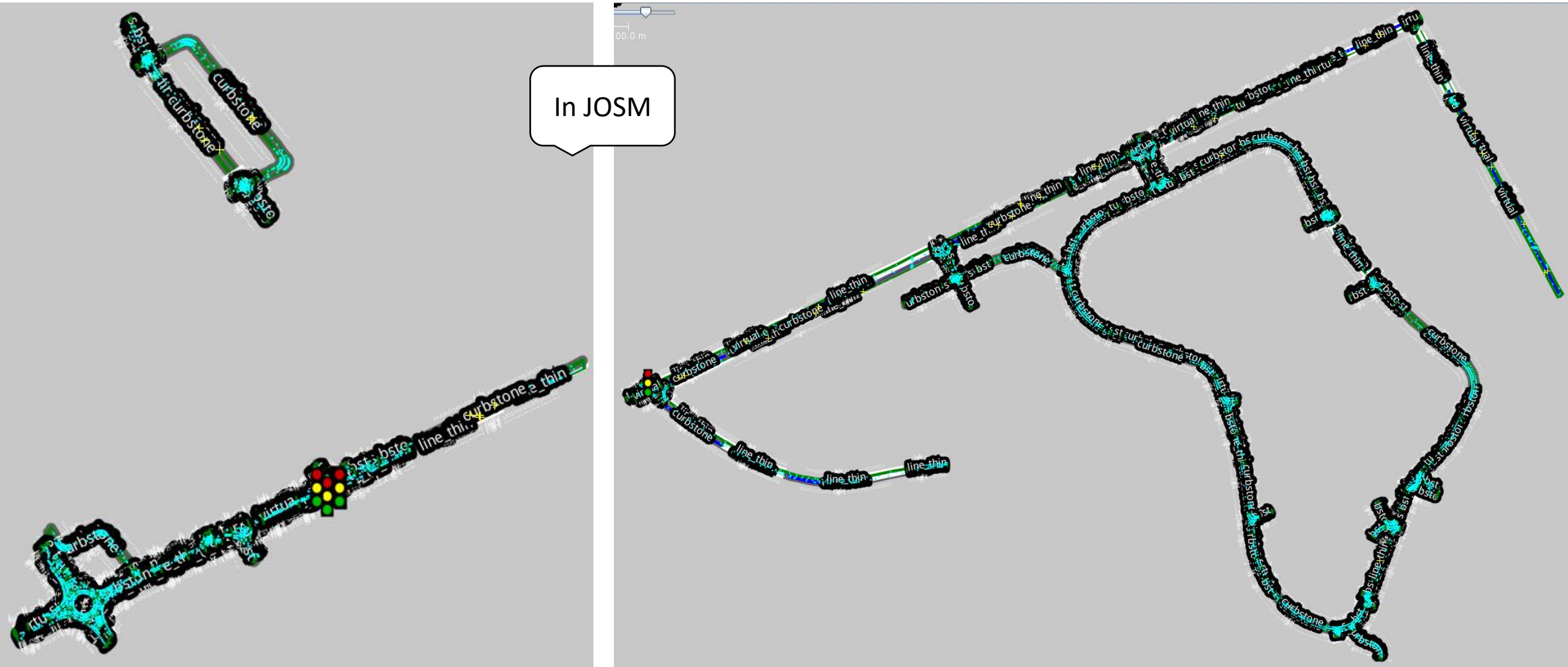
Traffic light controls a stop line



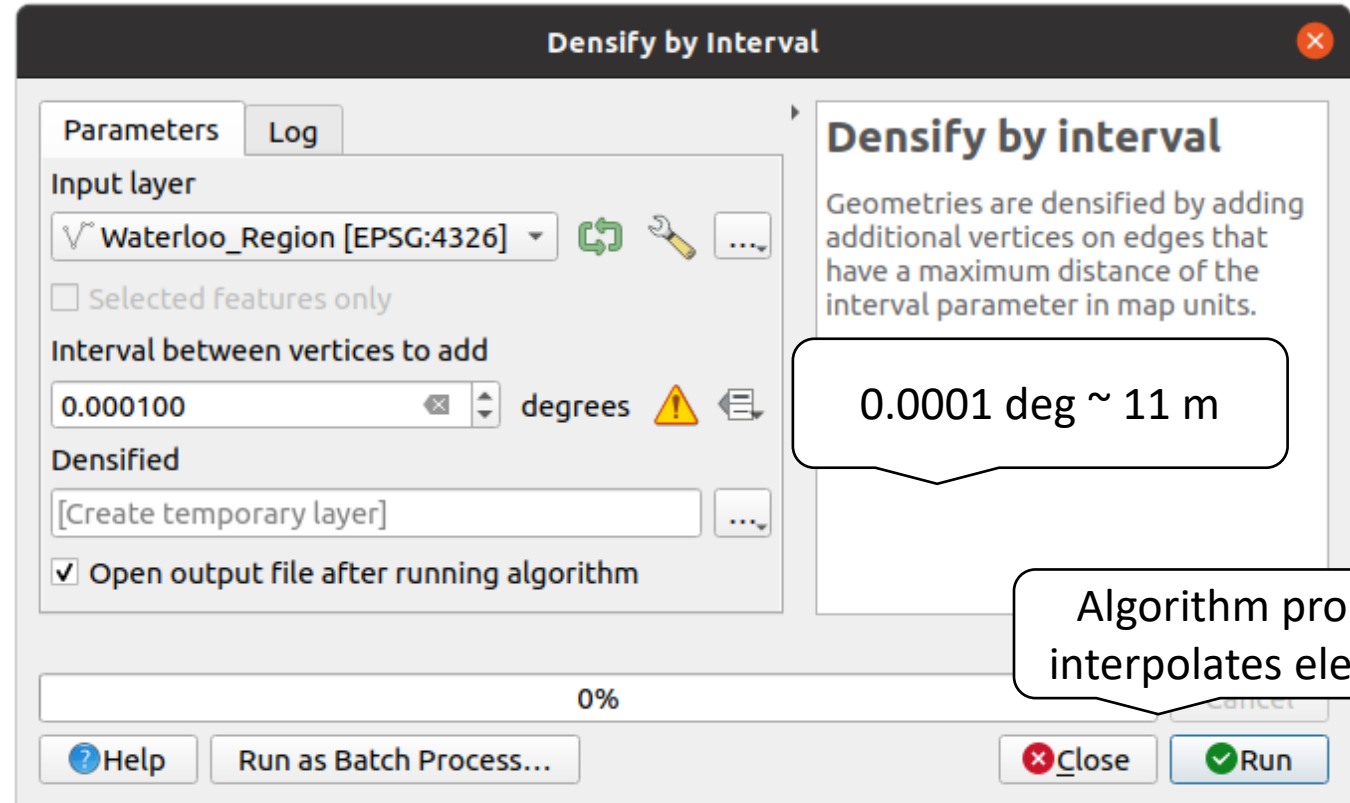
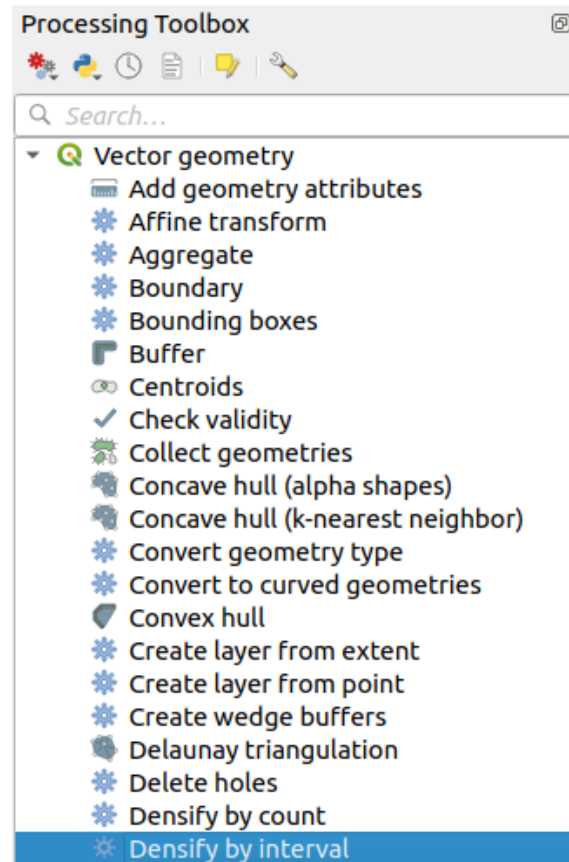
Feature	Value
Waterloo_Region	
catetitle	stop
(Derived)	
(Actions)	
fid	323500
catetitle	stop_line
rel	{"yielding": [323406, 323407]}

Stop line controls the yielding lanelets

1. Incrementally modifies the resulting Lanelet2 map



2. Map preparation using QGIS toolbox and our custom algorithms: densification



2. Map preparation using QGIS toolbox and our custom algorithms: elevation from GPS

The screenshot displays the QGIS interface with several key components:

- Layers Panel:** Located in the top-left, it shows a list of layers: 'gps_elevation' (checked), 'Waterloo_Region' (checked), 'Waterloo_Region_Imagery_2020' (checked), and 'Waterloo_Region_elevation' (unchecked).
- Map View:** The central area shows an aerial satellite image of a residential area. A network of roads is highlighted in black, and several red dots are scattered along these roads, representing collected GPS measurements.
- Vertex Editor:** A small window in the bottom-left shows a table with columns for 'x', 'y', and 'z' coordinates. The first row contains the values: -80.55405810, 43.45234389, and 315.79224...
- Python Console:** The bottom-center window shows the execution of a script. The command `exec(Path('/home/ma/AVIN-Mapping/avin-laneLet2/drape_from_gps.py')).read_text()` is entered, followed by `drape()`. The console displays progress bars for each step, from 0% to 100% (DONE).
- Python Editor:** The bottom-right window shows the source code for `drape_from_gps.py`. The code includes imports for `QgsFeature`, `QgsGeometry`, `QgsPoint`, `QgsProject`, `QgsSpatialIndex`, and `QgsWkbTypes` from the `qgis.core` module, and `iface` from `qgis.utils`. It also imports `time` and `numpy`.

Four callout boxes provide additional context:

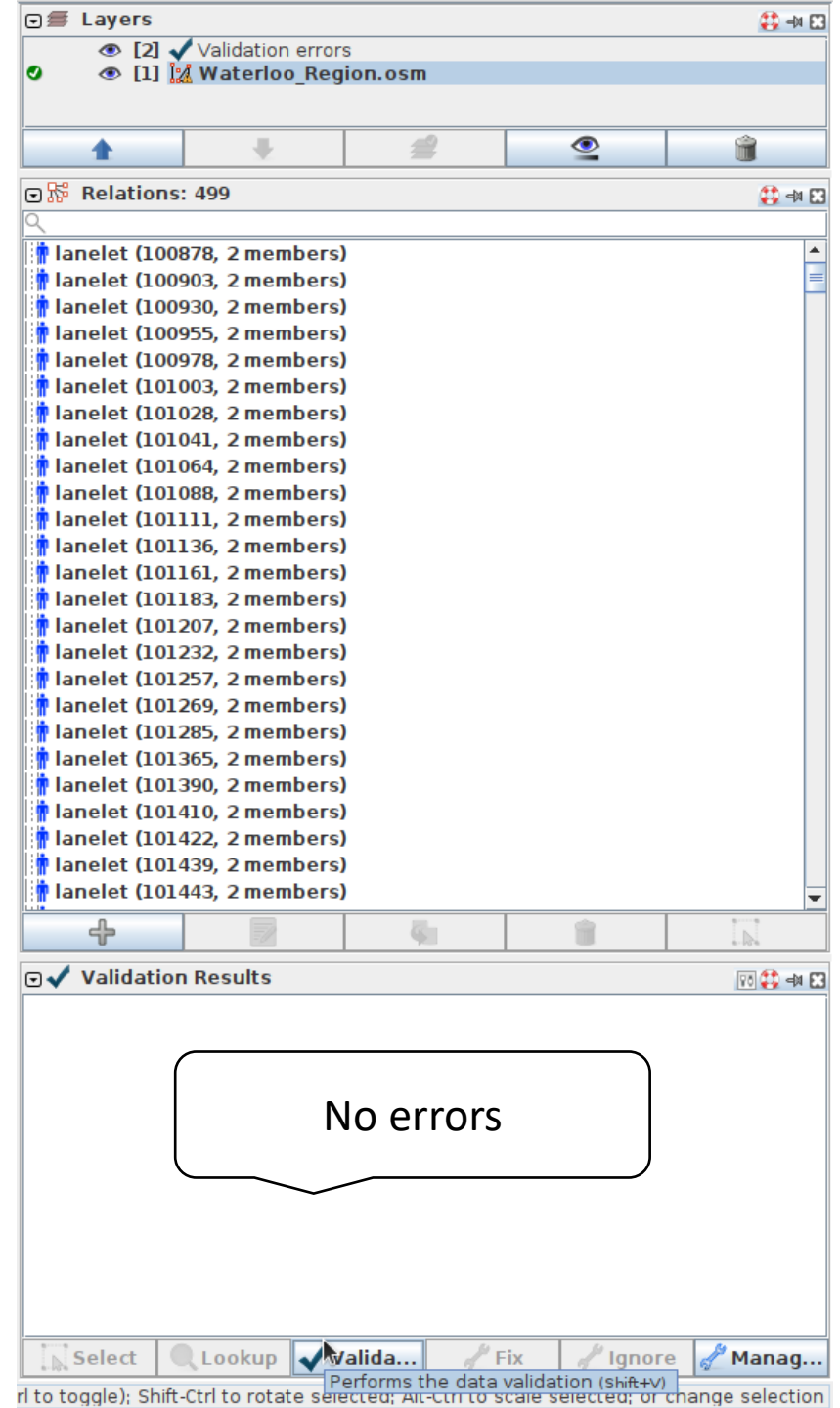
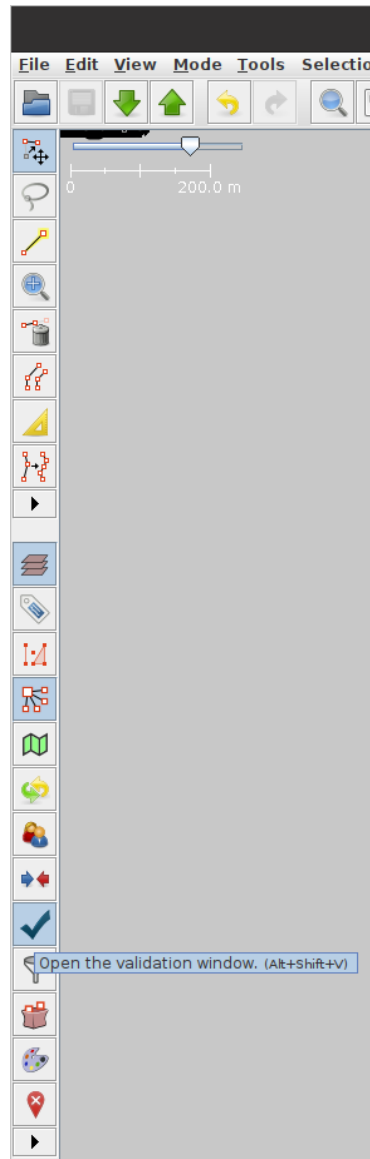
- Each measurement is a point with elevation:** Points to the red dots on the map.
- Collected GPS measurements:** Points to the black road network on the map.
- Execute drape function:** Points to the Python console showing the `drape()` command.
- Drape script uses GPS measurements to assign elevation in the map:** Points to the Python editor showing the `drape` class.

Phase 2: Map validation

Goal: "Validate the usefulness of the Lanelet2 map for automated driving systems"

- Demonstrated a layered four-step approach to validation
 1. Using JOSM
 - No duplicate nodes/ways nor other consistency issues
 2. Using "lanelet2_validate" from Lanelet2 library
 - Curvature, annotation, and other Lanelet2-specific issues
 3. Using GeoScenario Server
 - Map routing
 - Traffic simulations to exercise parts of the map
 4. Using WISE Automated Driving System (ADS)
 - Automated drive in WISE Sim simulation and on "UW Moose" vehicle

1. Validation using JOSM



2. Validation Using "lanelet2_validate" from Lanelet2 library

Points out defects in map design and quality

```
$ lanelet2_validate Waterloo_Region.osm --lat 43.4463235 --lon -80.5711598
Error: There is a 'left' relation from 102770 to 102079, but 102770 isn't the closest lanelet the other way round [routing.graph_is_valid]
Warning: lanelet 103759 Curvature at point 103752 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: lanelet 103548 Curvature at point 103547 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: lanelet 102506 Curvature at point 102500 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: lanelet 102506 Curvature at point 102501 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: lanelet 104116 Curvature at point 104075 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: lanelet 102612 Curvature at point 102600 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: lanelet 102612 Curvature at point 102601 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: lanelet 102612 Curvature at point 101536 is bigger than 0.5. This can confuse algorithms using this map. [mapping.curvature_too_big]
Warning: point 105869 Point is very close to point 101271. This can lead to defects in the map [mapping.points_too_close]
Warning: point 104621 Point is very close to point 104514. This can lead to defects in the map [mapping.points_too_close]
Warning: point 104514 Point is very close to point 104621. This can lead to defects in the map [mapping.points_too_close]
Warning: point 101271 Point is very close to point 105869. This can lead to defects in the map [mapping.points_too_close]
Warning: point 102320 Point is very close to point 102319. This can lead to defects in the map [mapping.points_too_close]
Warning: point 102319 Point is very close to point 102320. This can lead to defects in the map [mapping.points_too_close]
Warning: linestring 105916 attribute name is not a known lanelet2 tag. [mapping.unknown_tags]
Warning: linestring 105916 attribute qgis_fid is not a known lanelet2 tag. [mapping.unknown_tags]
Warning: linestring 105914 attribute qgis_fid is not a known lanelet2 tag. [mapping.unknown_tags]
```

Traffic light "name" needed for GeoScenario Server

"qgis_fid" needed for lanelet_utils script

3. Using GeoScenario Server

avin-lanelet2 scenario_suite

Name

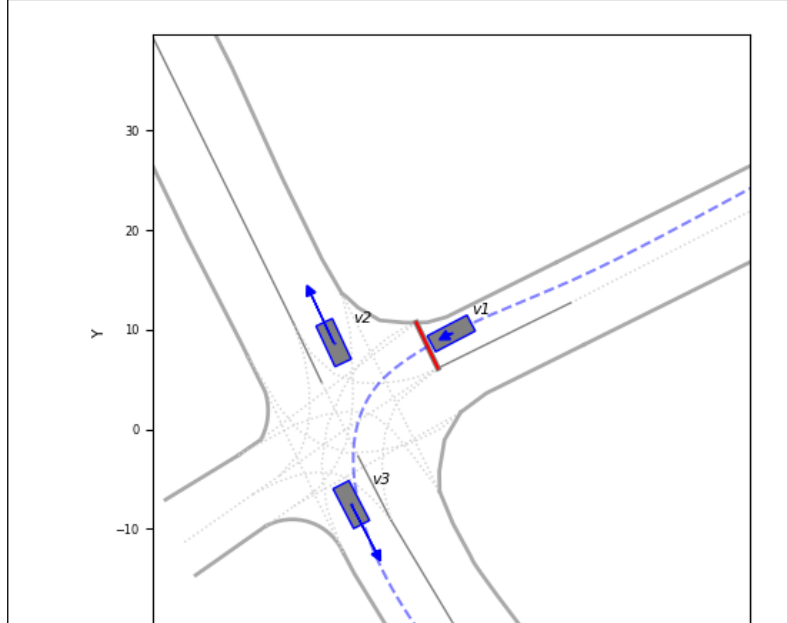
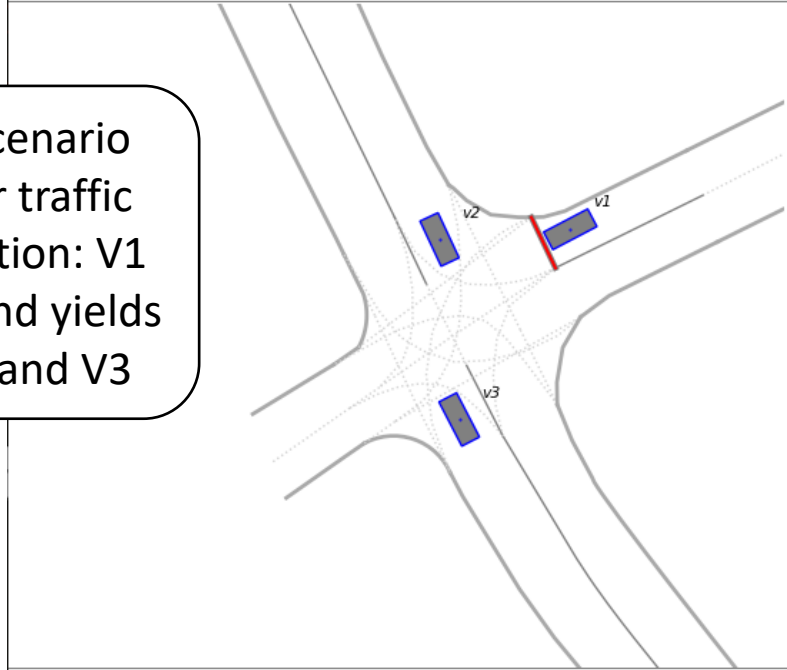
- geoscenarioserver
- log
- logs
- maps
 - Waterloo_Region
 - MapConfig.json
 - Waterloo_Region.osm
- scenarios
 - Erb-east-north
 - Erb-east-south
 - Erb-east-west
 - Erb-Erbsville-Ct-traffic-lights
 - Erb-north-east
 - Erb-south-east
 - Erb-south-south
 - Paradise-all-way-stop
 - Paradise-right-of-way
 - Ring-Road-ccw-west
 - Ring-Road-Columbia
 - Ring-Road-Will-Tutte
 - Westmount-Columbia
- scripts
 - launch.bash
 - scenario_suite_utilities.bash

A suite of scenarios

Launch script

```
$ bash scripts/launch.bash Erb-Erbsville-Ct-traffic-lights
Using current suite.
Starting GeoScenario server for /home/ma/AVIN-Mapping/avin-lanelet2/scenario_suite/scenarios/Erb-Erbsville-Ct-traffic-lights/Erb-Erbsville-Ct-traffic-lights.osm...
I0228 10:56:25.160545 32933 GSServer.py:21] GeoScenario server START
I0228 10:56:25.160643 32933 GSServer.py:32] Btree search locations set (in order) as: ['/home/ma/wise-sim-root/submodules/geoscenarioserver/btrees']
I0228 10:56:25.160696 32933 ScenarioSetup.py:33] Loading GeoScenario file(s): /home/ma/AVIN-Mapping/avin-lanelet2/scenario_suite/scenarios/Erb-Erbsville-Ct-traffic-lights/Erb-Erbsville-Ct-traffic-lights.osm
```

GeoScenario Server traffic simulation: V1 stops and yields to V2 and V3

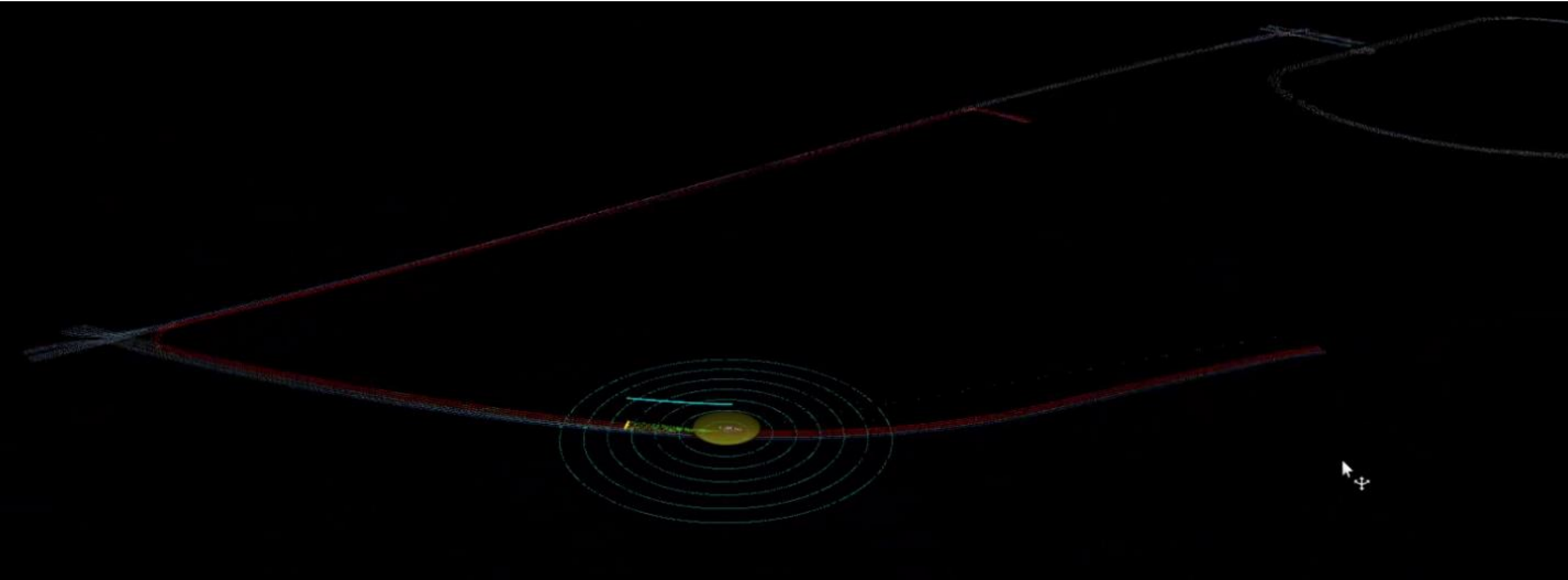


3. Using GeoScenario Server

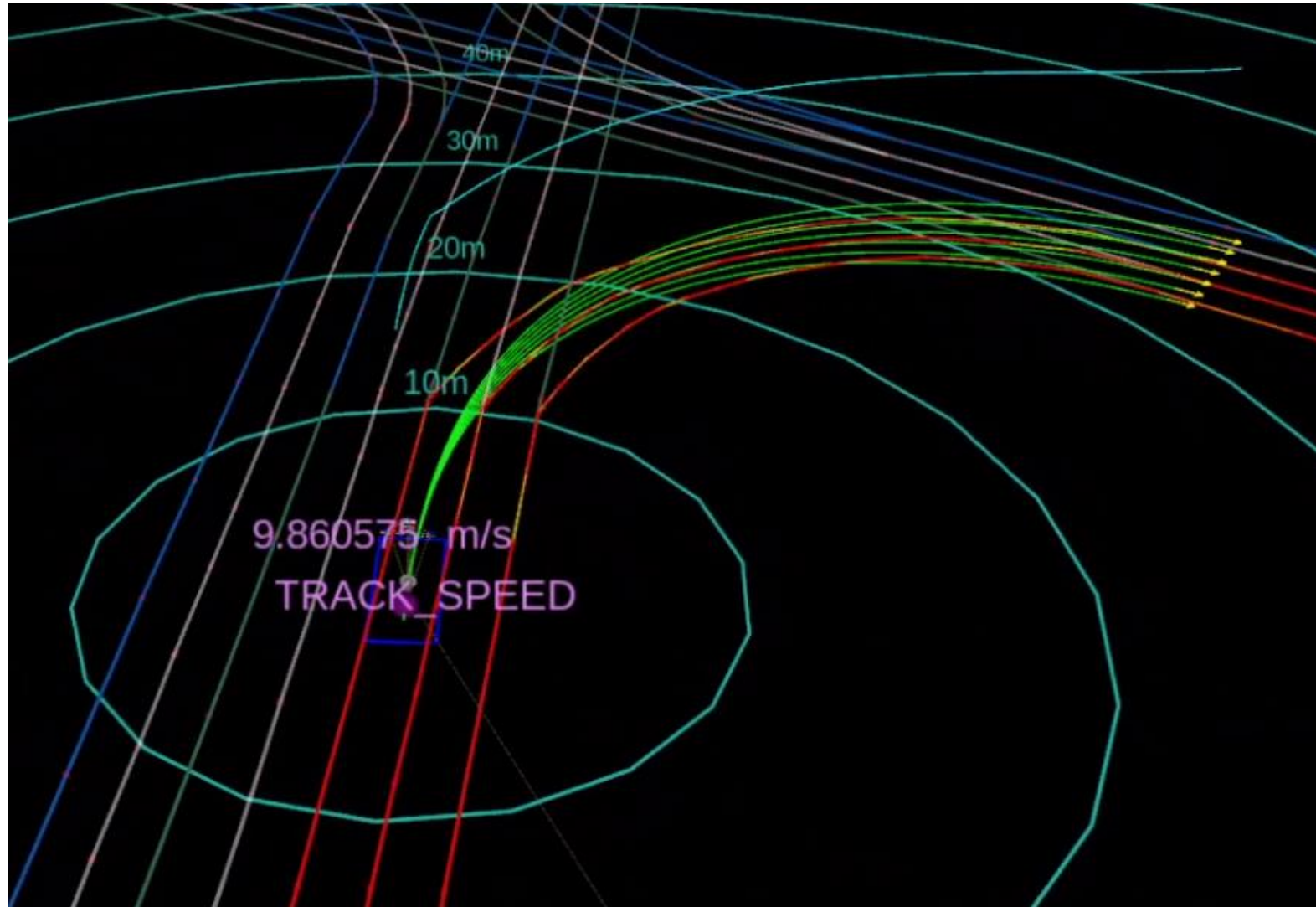
- Traffic simulation videos are available on WISE Lab's YouTube channel's playlist

<https://www.youtube.com/playlist?list=PL9WqTe3nDulh070tpdxsxZErboYloO8Yz>

4. Validation using WISE Automated Driving System (ADS): WISE Sim simulation



4. Validation using WISE Automated Driving System (ADS): WISE Sim simulation



4. Validation using WISE Automated Driving System (ADS): on "UW Moose"

- Tested on ["UW Moose" vehicle with WISE Automated Driving System \(ADS\)](#)
- WISE ADS does not handle traffic lights and lane changes
- Lanelet2 support within WISE ADS is not complete; however, the vehicle was able to drive on sections of the map

Open-source contributions

- Fixed numerous issues in the open-source Lanelet2 library
- Submitted two pull requests
 - <https://github.com/fzi-forschungszentrum-informatik/Lanelet2/pull/231>
 - <https://github.com/fzi-forschungszentrum-informatik/Lanelet2/pull/237>
- Added two new map projectors ("LocalCartesian" and "Geocentric")
 - Enable WISE ADS to drive on the map and eliminates displacements
 - Will submit a pull request
- Maintain a fork of the upstream repository until fixes merged
 - <https://github.com/wiselabuw/Lanelet2>

Thank You