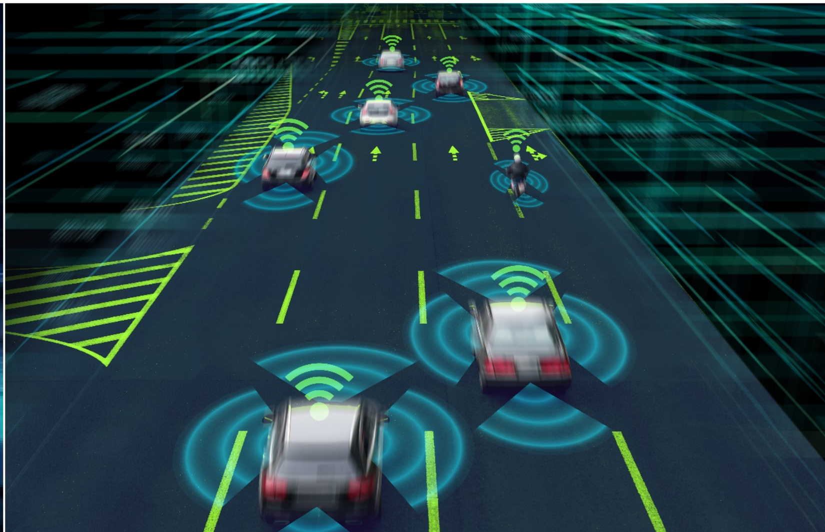
Leveraging OpenDRIVE and OpenSCENARIO to connect inter-disciplinary teams for automated driving

John Pallag
jpallag@mathworks.com
RoadRunner Products
Senior Developer

Mark Corless
mcorless@mathworks.com
Automated Driving
Segment Manager

Avinash Nehemiah
anehemia@mathworks.com
Autonomous Systems & AI
Product Manager



Outline

- Quick company introduction
- **Case Study:** Generating OpenSCENARIO from recorded sensor data
- **Engineering Insight:** Preserving Data Fidelity with OpenDRIVE

Headquarters

Natick, MA USA

North America

United States

Europe

Finland
France
Germany
Ireland
Italy
Netherlands
Spain
Sweden
Switzerland
UK

Asia-Pacific

Australia
China
India
Japan
Korea

5000+ staff

in 34 offices around the world

$1+ billion

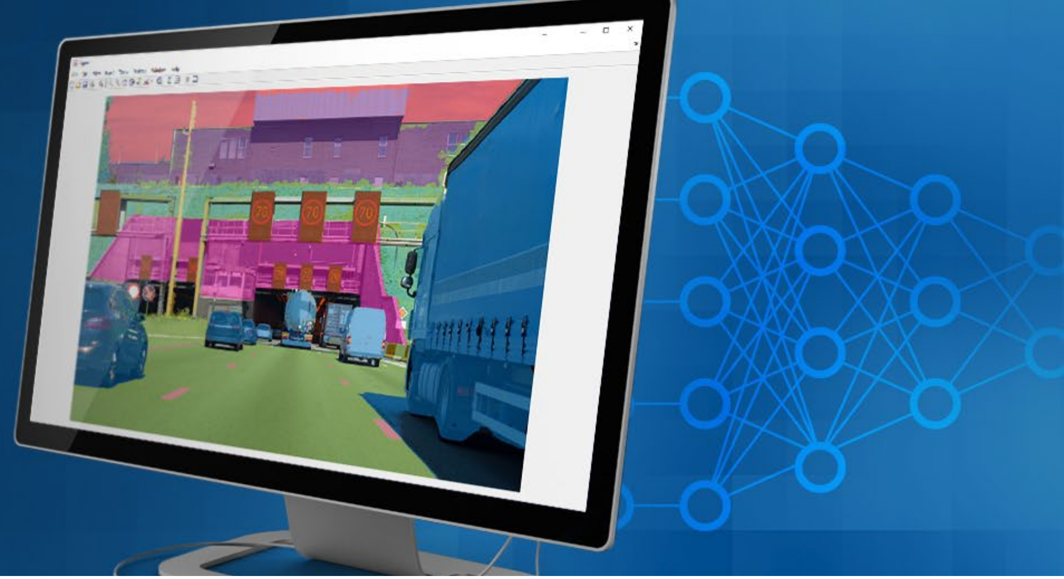
in revenues

Privately held

and profitable every year

Our Products

MATLAB® & SIMULINK®

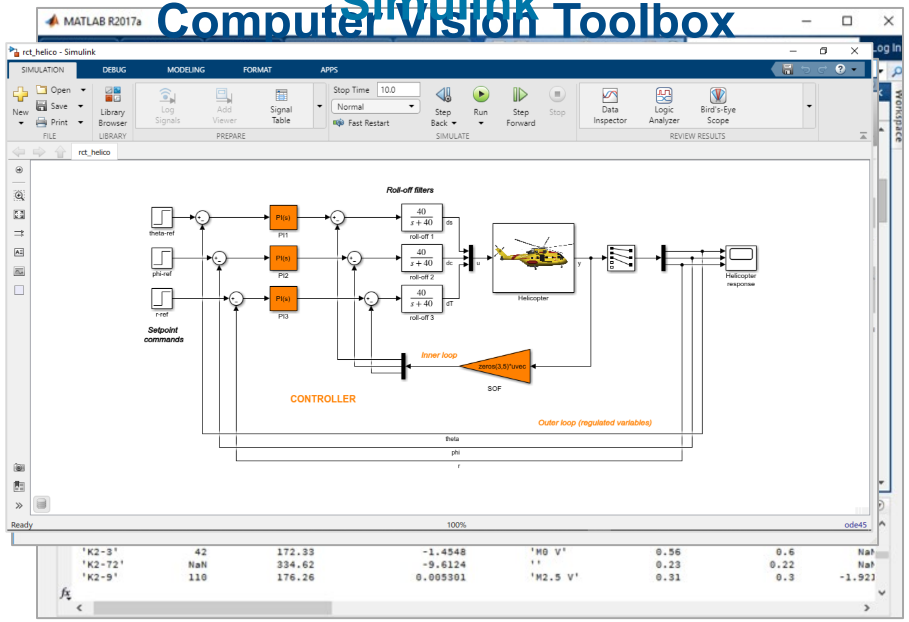


- MATLAB is a programming environment for algorithm development, data analysis, visualization, and numeric computation.
- Simulink is a graphical environment for designing, simulating, and testing systems.
- More than 100 add-on products for specialized tasks.

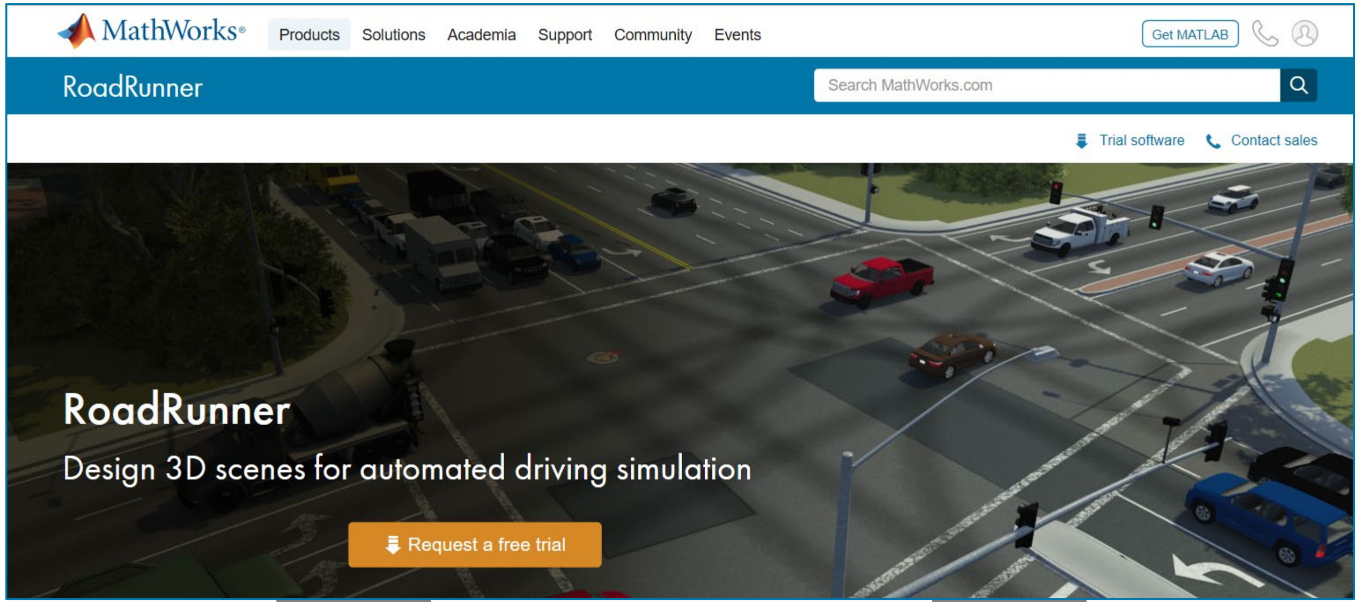
MATLAB

Simulink

Computer Vision Toolbox



Design 3D scenes for automated driving simulation

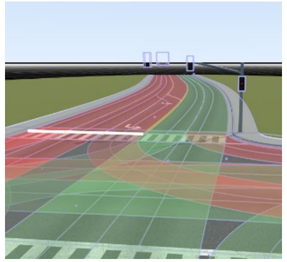


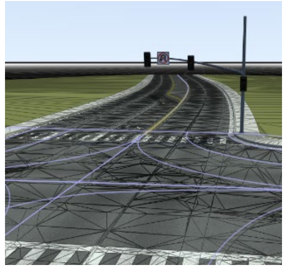
R2020a
Update 1

New base product

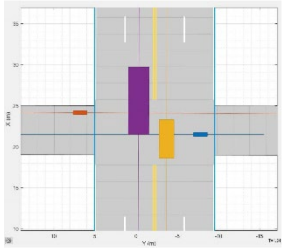
Does not require MATLAB

External Simulators



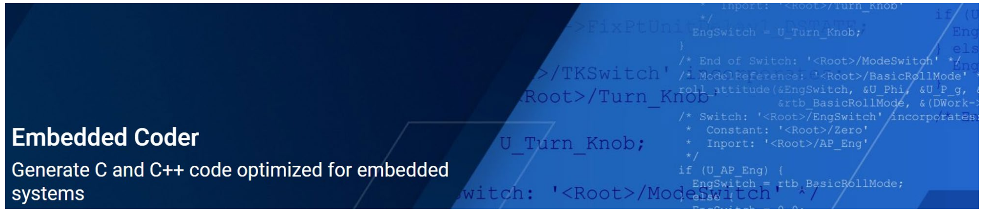


MATLAB & Simulink



ASAM standards are supported across MathWorks products

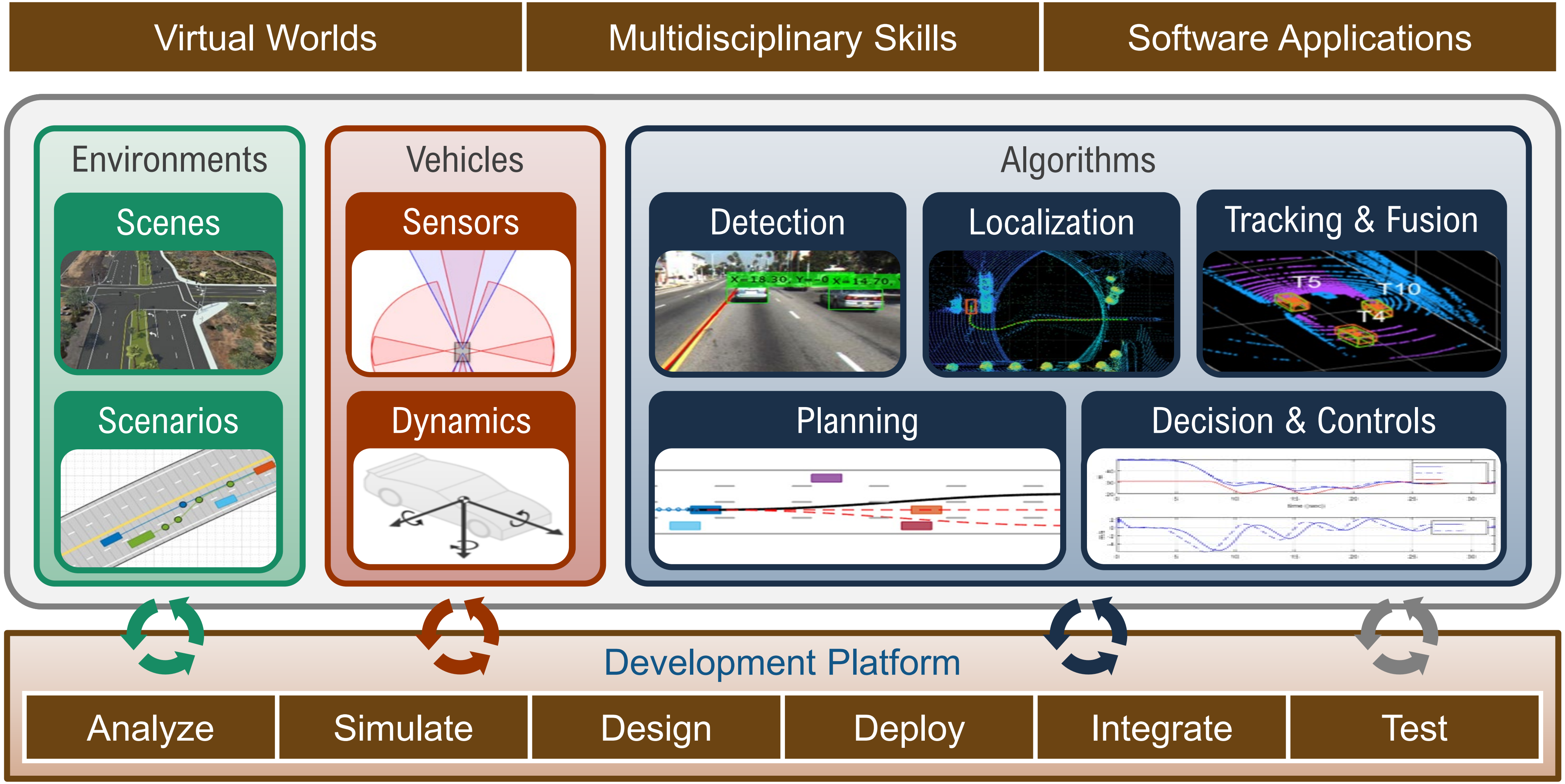
- Automated Driving
  - OpenDRIVE
  - OpenSCENARIO
  - OpenCRG
- Vehicle Networking and Communication
  - XCP A2L
  - MDF
- Calibration and Measurement
  - MCD-2 MC
  - AE CDF
- Rapid Prototyping
  - XIL API



Outline

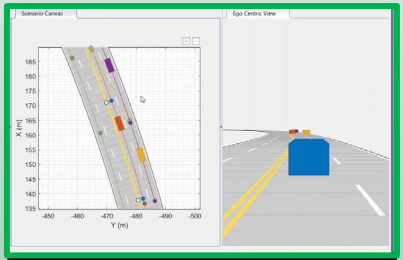
- Quick company introduction
- **Case Study: Generating OpenSCENARIO from recorded sensor data**
- **Engineering Insight:** Preserving Data Fidelity with OpenDRIVE

Develop Automated Driving Systems
with MATLAB, Simulink, RoadRunner, and Polyspace



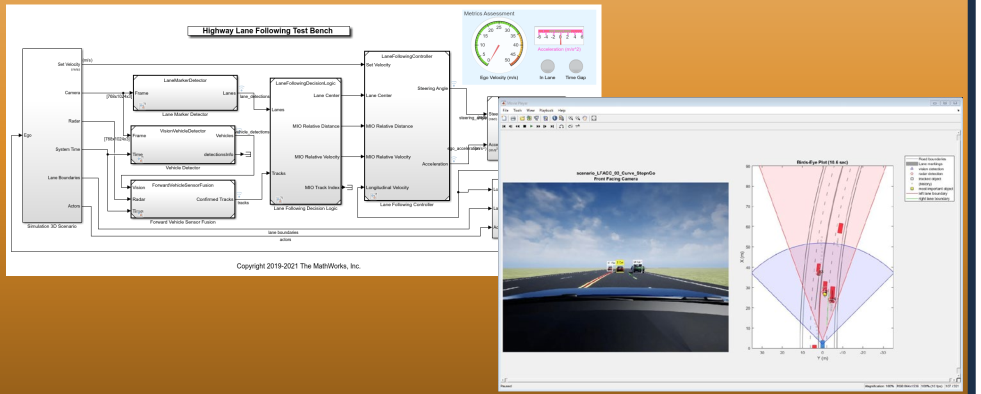
ASAM standards enable reuse of scenes and scenario across multiple driving simulation environments

Author Scenes & Scenarios



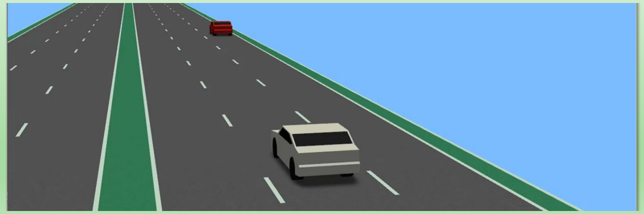
drivingScenario

Simulate with MATLAB & Simulink



OpenDRIVE & OpenSCENARIO

Simulate with Other Driving Simulation Environments



GM synthesizes scenarios from recorded data to validate lane centering system

Scenes | Sensors | Detection | Localization | Tracking
Scenarios | Dynamics | Planning | Controls
Analyze | Simulate | Design | Deploy | Integrate | Test

2020-01-0718 Published 14 Apr 2020

SAE INTERNATIONAL

Creating Driving Scenarios from Recorded Vehicle Data for Validating Lane Centering System in Highway Traffic

Seo-Wook Park, Kunal Patil, Will Wilson, and Mark Corless The MathWorks, Inc.

Gabriel Choi and Paul Adam General Motors LLC

Citation: Park, S.-W., Patil, K., Wilson, W., Corless, M. et al., "Creating Driving Scenarios from Recorded Vehicle Data fo[r] ... Centering System in Highway Traffic," SAE Technical Paper 2020-01-0718, 2020, doi:10.4271/2020-01-0718.

The methodology to create a virtual driving scenario consists of the following steps:

1. Record and select data
2. Reconstruct road network
3. Localize ego trajectory
4. Reconstruct target vehicles
5. Compare with recorded video

The virtual driving scenario can then be used to test an ADAS system using simulation. This paper demonstrates applying this methodology to test a lane centering application.

FIGURE 14 Driving case (c): cut-in vehicle with too close distance

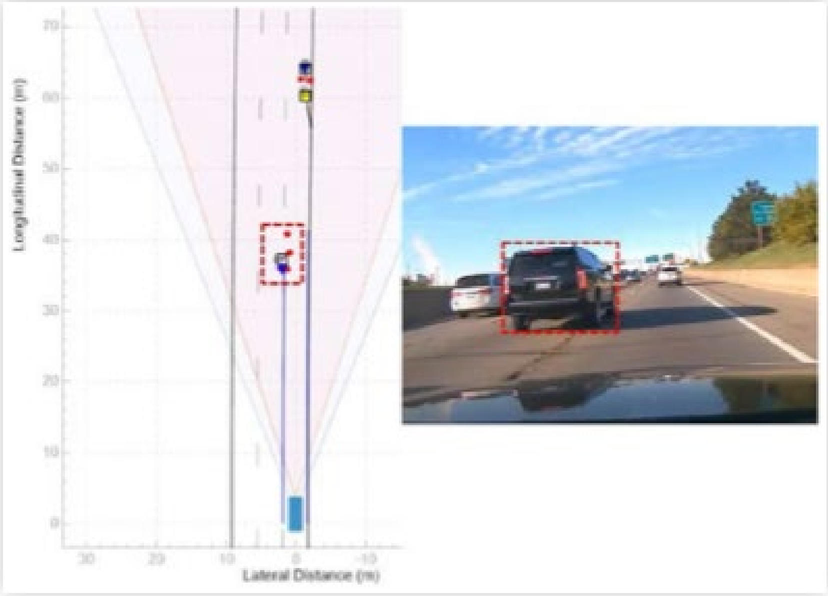
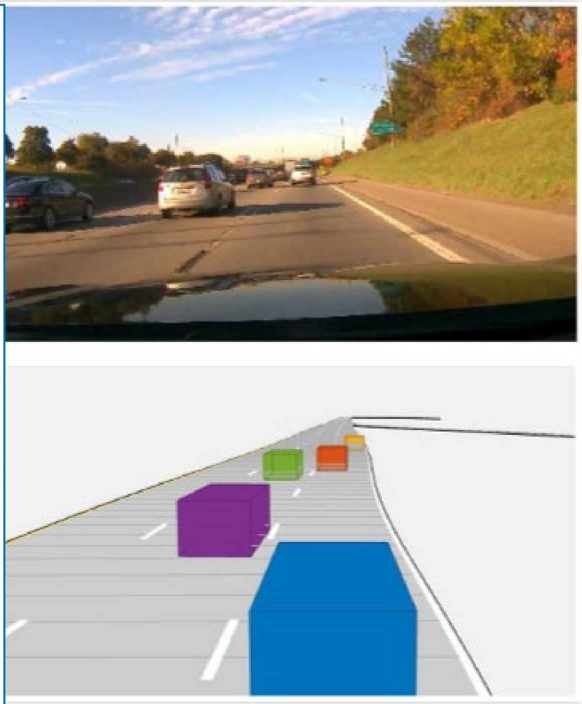
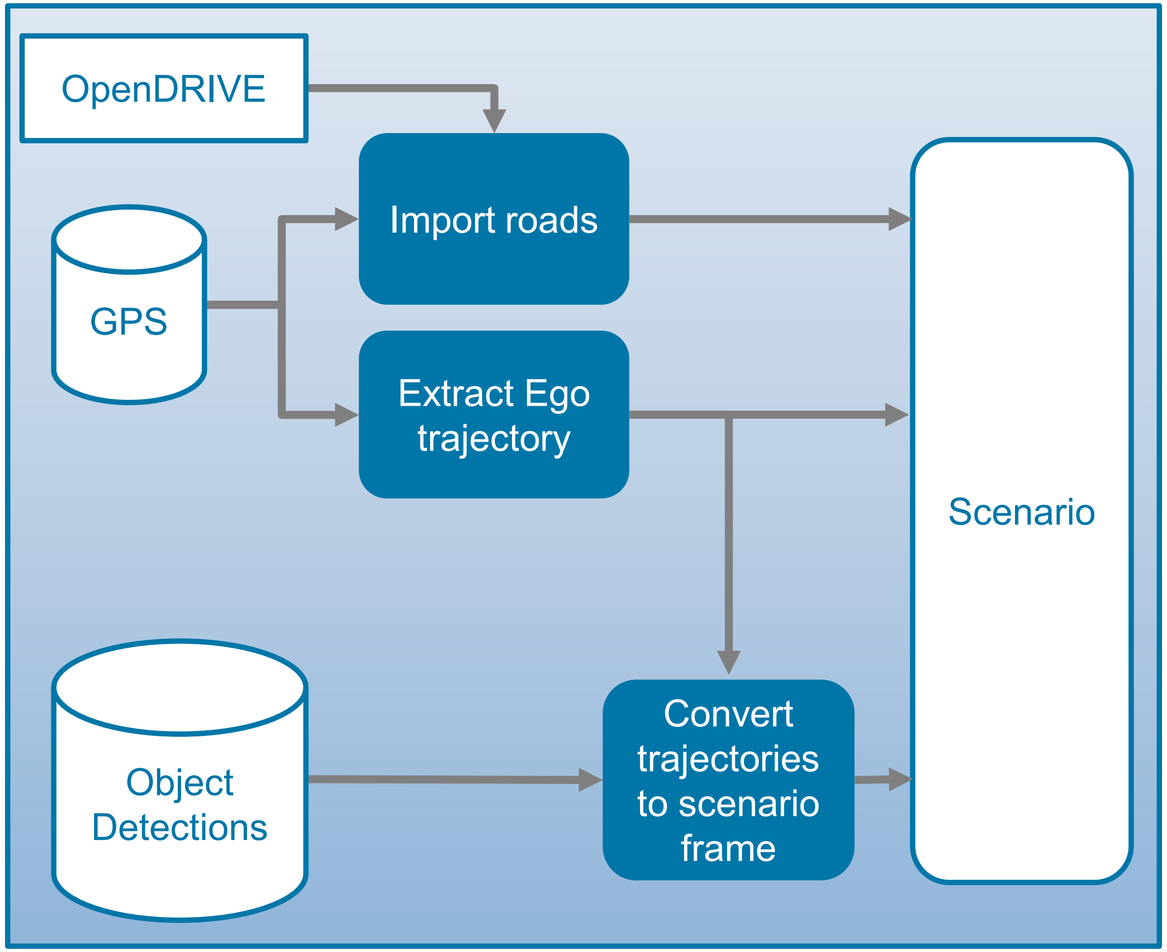


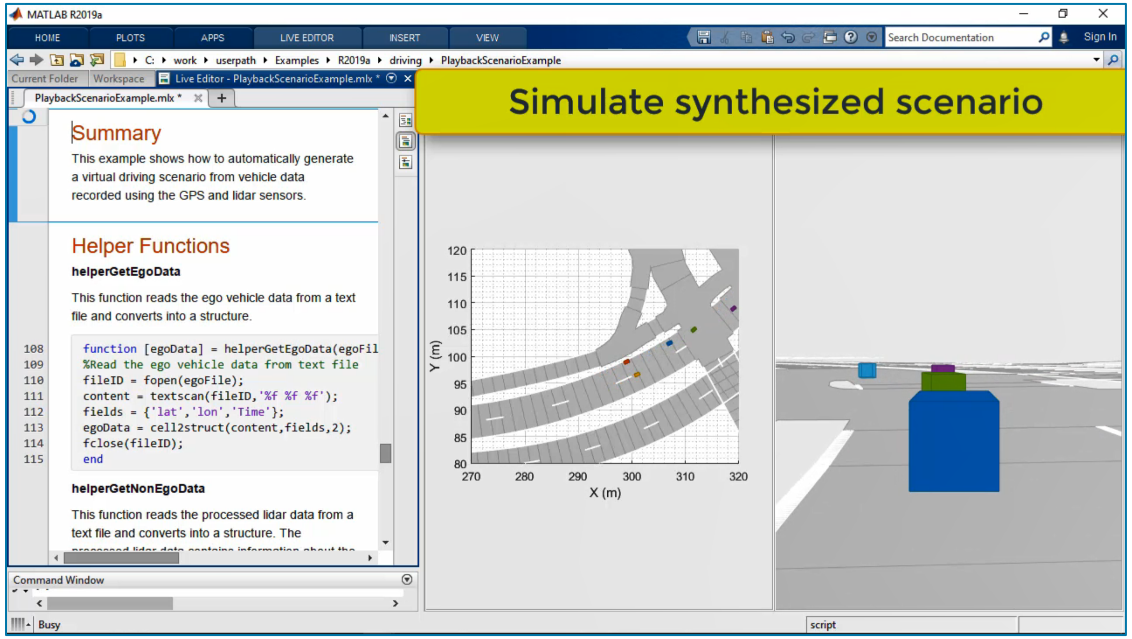
FIGURE 9 Recorded video (top) vs. reconstructed driving scenario (bottom)



SAE Paper 2020-01-0718 (technical article)
2020 MathWorks Automotive Conference Presentation (recording, slides)

Generate scenario from recorded GPS and object detections





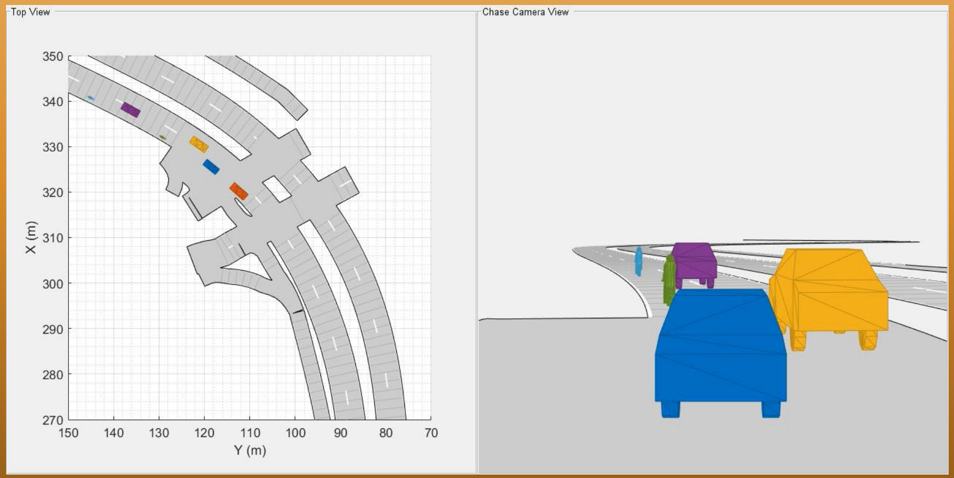
- Ego trajectories are extracted from GPS
- Non-Ego trajectories are extracted from object detection list
- Camera is used for visual verification

[Scenario Generation from Recorded Vehicle Data]

Automated Driving Toolbox™

Generate scenarios in MATLAB and play in ESMINI

Export to OpenSCENARIO from MATLAB



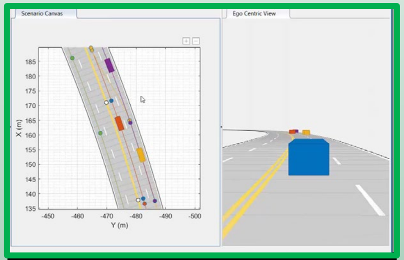
Import OpenSCENARIO to ESMINI

```
%% Export driving scenario to OpenSCENARIO
oscFilename = "playbackScenario_OSC.xml"
export(scenario "OpenSCENARIO",oscFilename);
```

```
esmini
--camera_mode driver
--road_features off
--osc "playbackScenario_OSC.xml"
```

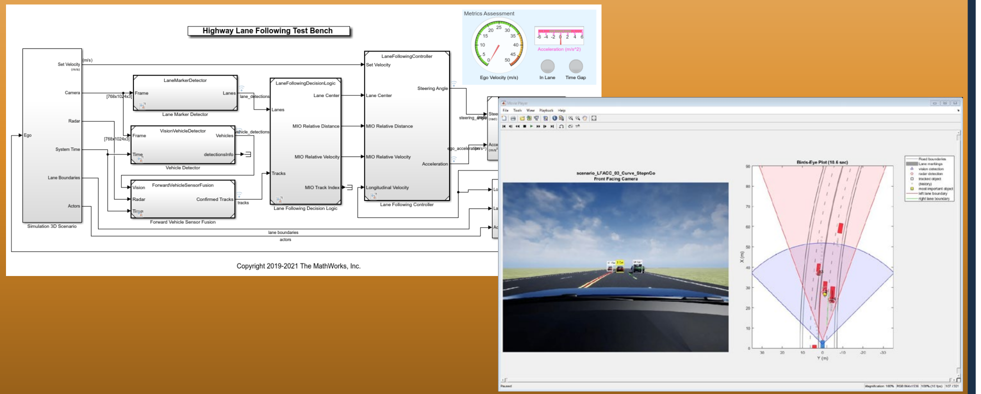
ASAM standards enable reuse of scenes and scenario across multiple driving simulation environments

Author Scenes & Scenarios



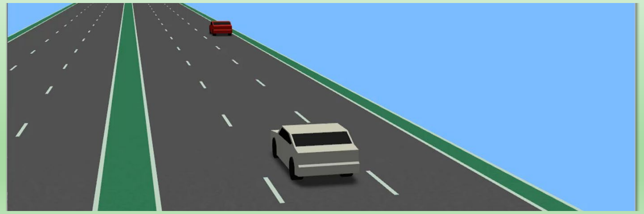
drivingScenario

Simulate with
MATLAB & Simulink



OpenDRIVE &
OpenSCENARIO

Simulate with Other
Driving Simulation Environments



Outline

- Quick company introduction
- **Case Study:** Generating OpenSCENARIO from recorded sensor data
- <u>**Engineering Insight: Preserving Data Fidelity with OpenDRIVE**</u>

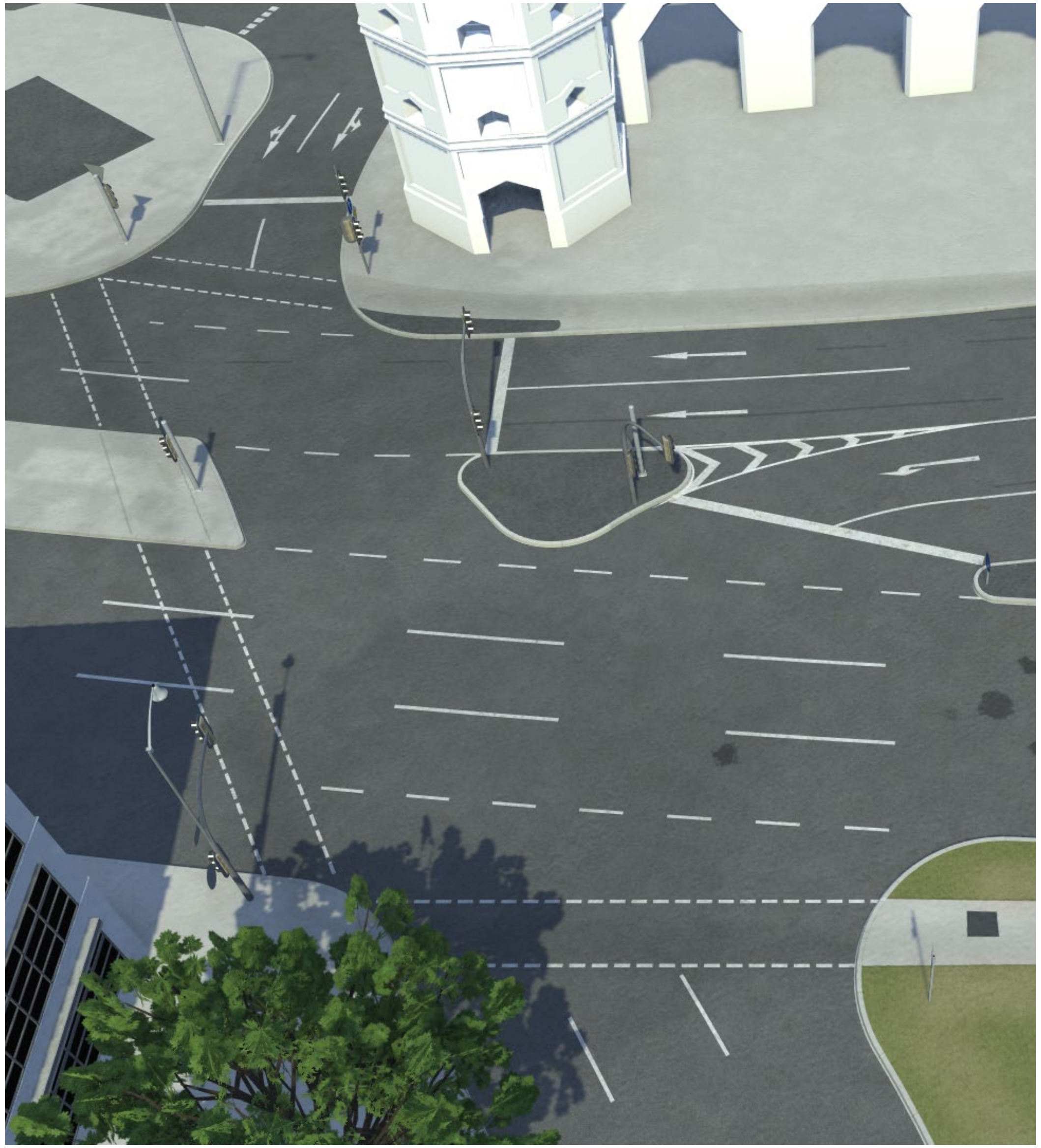
Preserving Data Fidelity with OpenDRIVE

John Pallag
MathWorks

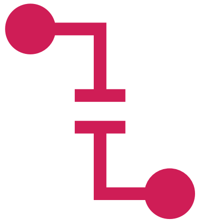
Introduction

- John Pallag
- RoadRunner/VectorZero
- MathWorks RoadRunner team

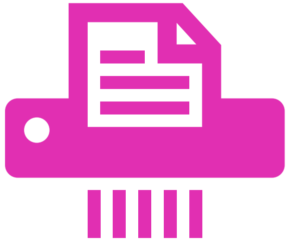
- OpenDRIVE
- Road Editing
- HD Maps



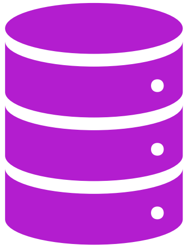
Data Fidelity



Determining "how close" the resulting data lies to the source data



After performing operations on the source, how similar is the resulting file (ex. Load -> save)



Does the data still contain usable information?

Why is data fidelity important? (besides the obvious)

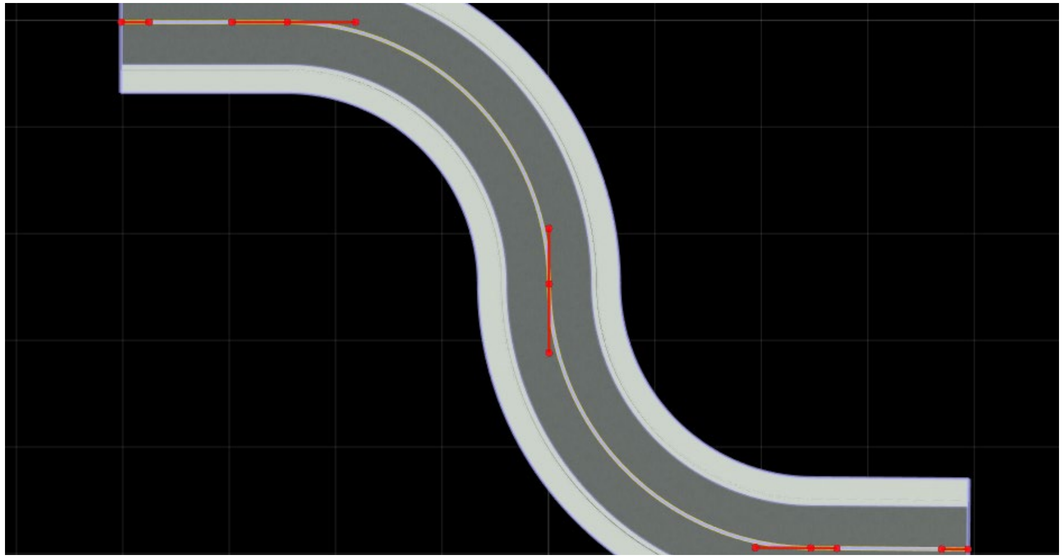
Real-world accuracy

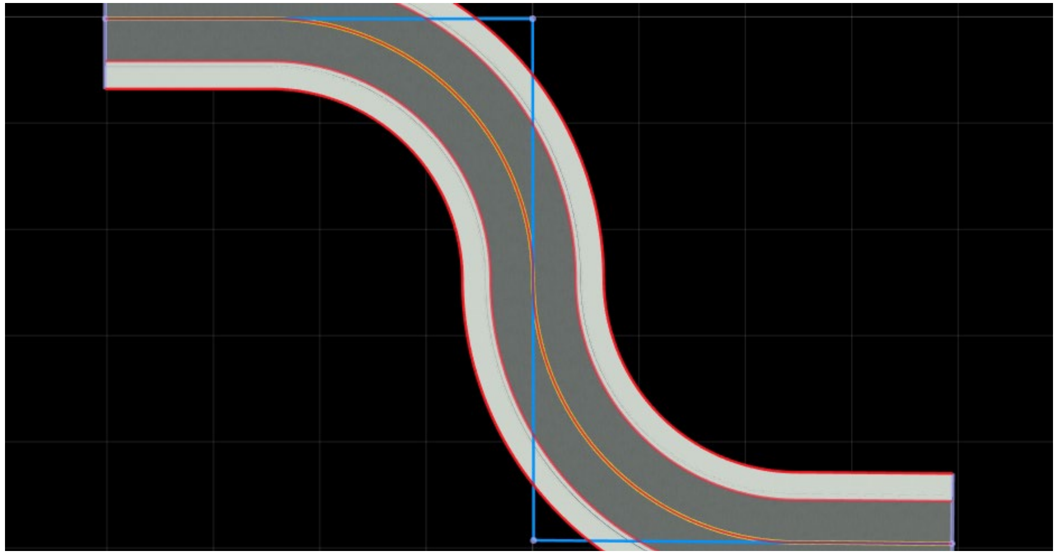
Internal formats

Real-world accuracy

- Current OpenDRIVE:
  - OpenDRIVE 1.5+ <dataQuality>
  - GIS data with the same projection
- Future:
  - Working with data from different projections
  - Working in software with different projections

The internal format





Editing OpenDRIVE is tricky

- Adjusting the start of a slip road or on/off ramp
- Adjusting the start of a lane

Simulating on OpenDRIVE is tricky

- Complex curve and offset curve computations
- Complex elevation model

- 1. Import OpenDRIVE
- 2. Make changes to internal scene/scenario
- 3. Export OpenDRIVE (same?)

Areas of fidelity

- Geometry
- Id/Name
- Types/Subtypes
- UserData

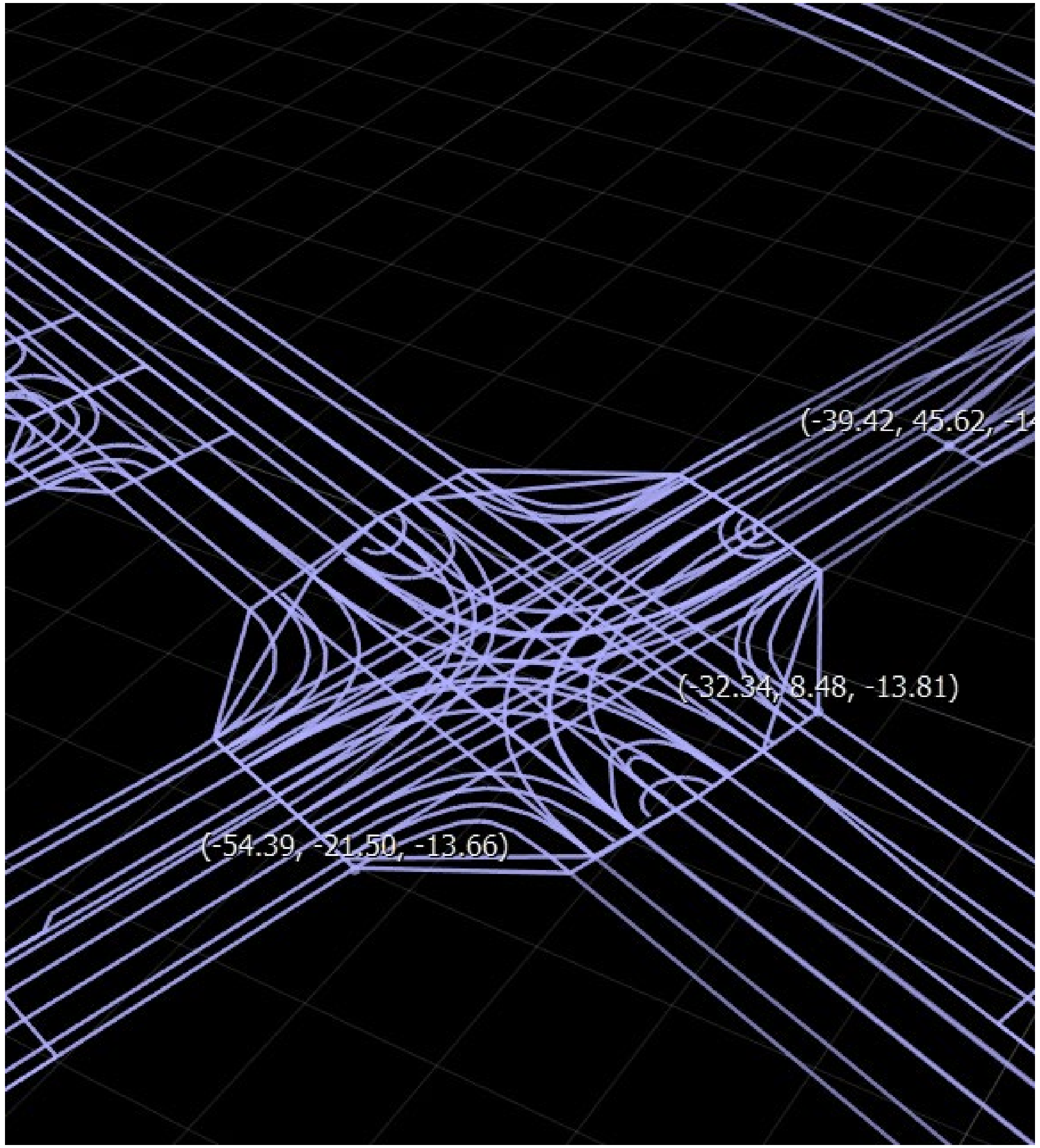
Geometric fidelity

Same projection:

- "Are the numbers the same?"
- Testing points along lanes/reference road
- Area Model: Points match

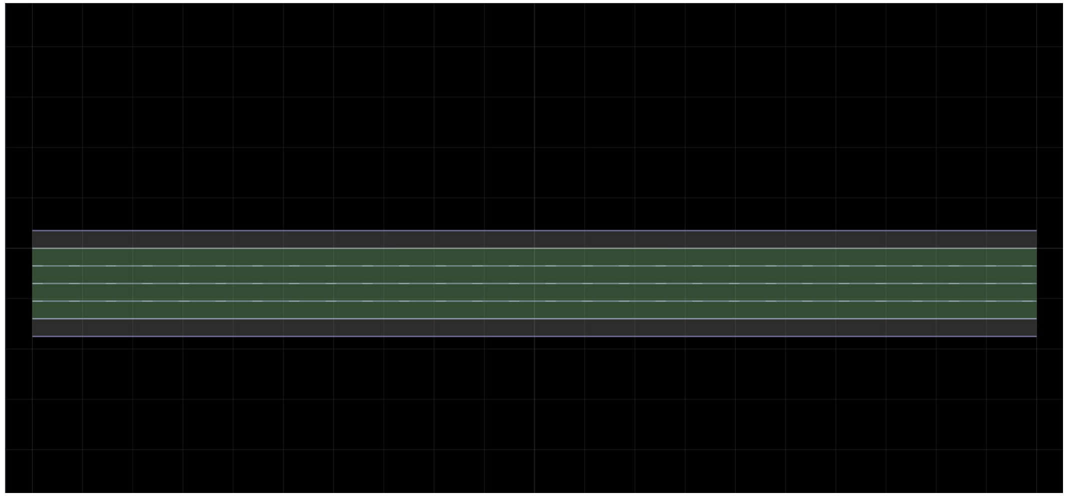
Different projection?:

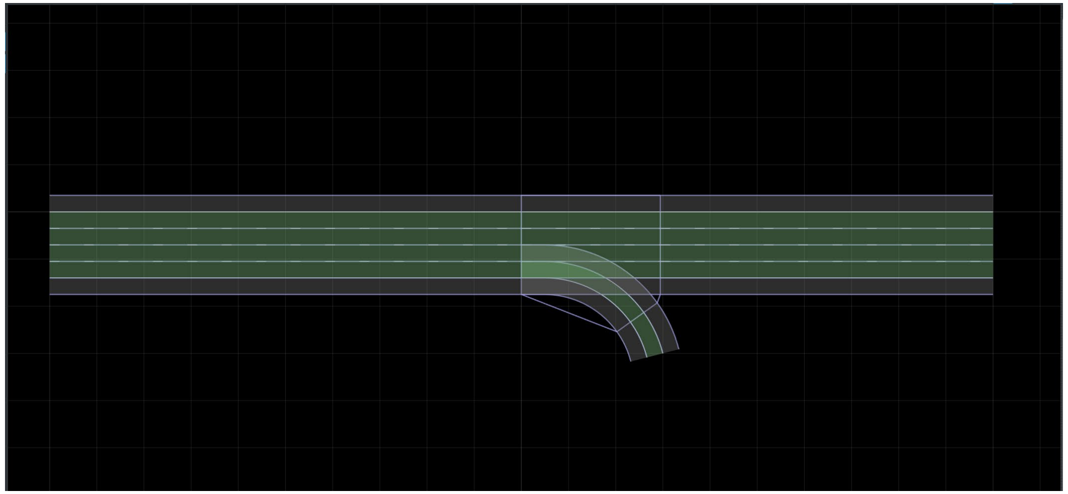
- Testing points have the same lat/long
- Testing geometry still has no "leaps" or "kinks"
- Area Model: Exported points have the same lat/long



Id/Name fidelity

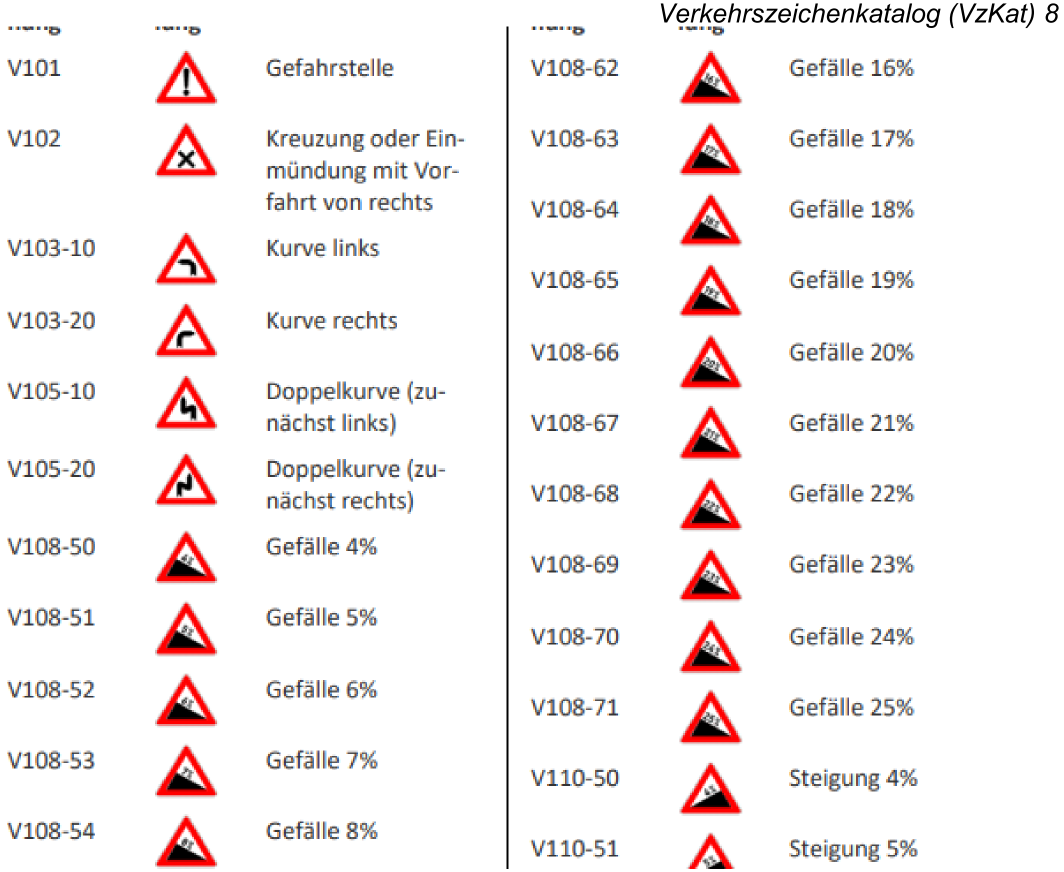
- "Are my ids still correct?"
- Ensure software "down the pipeline" can use the data
- Common Example:
  Add an off ramp to a stretch of highway represented as one <road> in OpenDRIVE





Type fidelity

- Already progress in upcoming OpenDRIVE versions
- A common understanding of signal/sign/object types
- Maintaining signals/signs/objects that may be unused


Verkehrszeichenkatalog (VzKat) 8

V101		Gefahrstelle	V108-62		Gefälle 16%
V102		Kreuzung oder Einmündung mit Vorfahrt von rechts	V108-63		Gefälle 17%
			V108-64		Gefälle 18%
V103-10		Kurve links	V108-65		Gefälle 19%
V103-20		Kurve rechts	V108-66		Gefälle 20%
V105-10		Doppelkurve (zunächst links)	V108-67		Gefälle 21%
V105-20		Doppelkurve (zunächst rechts)	V108-68		Gefälle 22%
V108-50		Gefälle 4%	V108-69		Gefälle 23%
V108-51		Gefälle 5%	V108-70		Gefälle 24%
V108-52		Gefälle 6%	V108-71		Gefälle 25%
V108-53		Gefälle 7%	V110-50		Steigung 4%
V108-54		Gefälle 8%	V110-51		Steigung 5%

User Data fidelity

- OpenDRIVE <userData>
- Can be important for pipelines
- Can carry information from source data
- Common data currently stored in <userData>

Conclusion

- Software in a pipeline has different needs of the data
- How can the format help ensure the important data remains through the pipeline?
- How can the format encourage usage standardization?
- How can OpenDRIVE grow to better represent real-world data and "just work" with other GIS data?