High-performance serving of large-scale OpenDRIVE datasets using standardized GIS technology

6th Symposium Driving Simulation, 2020-11-05, Virtual Event

Michael Scholz
German Aerospace Center (DLR)

Institute of Transportation Systems
German Aerospace Center

Research institutes
- Aeronautics
- Space
- Energy
- Transportation
- Security
- Digitalisation

Space administration
Project management agency
Institute of Transportation Systems

Key facts
• In Berlin and Brunswick
• Around 220 employees

Research fields
• Automotive
• Railway systems
• Traffic management
• Multi-modal and public transport

Area of work
• Fundamental research
• Conception and strategy development
• Prototyping
Our research infrastructure ...
... and our Testbed of Lower Saxony
Challenges
Testbed Lower Saxony – Part 1
What is desirable?

• Consistent management of such heterogeneous datasets

• Fast access and data browsing for different users

• Simple interface/API

• Easy snippet extraction from whole datasets
OpenDRIVE over time

<table>
<thead>
<tr>
<th>Initial scope of application</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Fast prototyping of simulation tracks → Artificial/imaginary test data</td>
</tr>
<tr>
<td>• Restricted (small) spatial extent</td>
</tr>
<tr>
<td>• High modelling detail with visual properties → 3D rendering</td>
</tr>
<tr>
<td>• Simple, continuous geometry definition → Smooth road course</td>
</tr>
<tr>
<td>• Real-time processing capability</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Current and future trends</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Real-world data</td>
</tr>
<tr>
<td>• From motorways over</td>
</tr>
<tr>
<td>• inner cities to</td>
</tr>
<tr>
<td>• multi-level parking decks</td>
</tr>
<tr>
<td>• Data updates and network merging</td>
</tr>
<tr>
<td>• Increasing spatial extents</td>
</tr>
<tr>
<td>• “From simulation into the car” → Electronic horizon, rejection of styling elements</td>
</tr>
<tr>
<td>• Linkage to supplementary environmental data</td>
</tr>
</tbody>
</table>
OpenDRIVE is bad for

Extracting/editing

• Because you need tools which are
  • commercial (money, money, money),
  • complicated,
  • inflexible,
  • ugly.

• Because there is no server-based solution “as a service”

see ASAM OpenDRIVE Format Specification, v. 1.6.0
OpenDRIVE is bad for

Data exchange

• Because of strong data model hierarchy and element cross-references

```xml
<road name="Boulevard of Rock" length="66.6">
  <link>
    <predecessor elementType="junction" />
    <successor elementType="junction" />
  </link>
</road>

<signal s="0" t="0" id="1337" country="LV-426" subtype="-1">
  <laneValidity fromLane="1">
  </laneValidity>
</signal>

<junction name="ne Kreuzung halt" id="1234">
  <connection id="0" incomingRoad="1" connectingRoad="2">
    <laneLink from="-7" to="-7"/>
    <laneLink from="-6" to="-6"/>
    <laneLink from="-5" to="-5"/>
    <laneLink from="-4" to="-4"/>
  </connection>
</junction>

<lanes>
  <laneSection s="0">
    <left>
      <lane id="3" type="border">
        <link>
          <successor id="3" />
        </link>
      </lane>
    </left>
  </laneSection>
</lanes>
```
OpenDRIVE is bad for

Data exchange

- Because of strong data model hierarchy and element cross-references
- Because small data snippets quickly result in millions of lines of text

![Code snippet from an XML file showing the length and number of lines of text.](image-url)
Subset extraction is not trivial
Solving “the problem” in three steps

1. Make OpenDRIVE data GIS-able
2. Deploy GIS data in spatial database
3. Publish as RESTful web service
GIS? Why?

• Well-established standards around for 15+ years (OGC)

• A super-huge community of which all automotive guys can just dream about

• They know how to handle huge data

• Broad tool support, also for free (open source software)

• Some components offer standardised services

• Native workflow with cadastral data, CAD, Road2Simulation, Lanelet/2, …
GIS? Why?

- Well-established **standards** around for 15+ years (OGC)
- A super-huge **community** of which all automotive guys can just dream about
- They know how to **handle huge data**
- Broad tool support, also **for free** (open source software)
- Some components offer **standardised services**
- Native workflow with cadastral data, CAD, Road2Simulation, Lanelet/2, …

“Don’t re-invent the wheel”
Solving “the problem” in three steps

1. Make OpenDRIVE data GIS-able
2. Deploy GIS data in spatial database
3. Publish as RESTful web service
Make OpenDRIVE data GIS-able
Geometry basics

• Elements refer to an imaginary reference line
Make OpenDRIVE data GIS-able

Geometry basics

- Elements refer to an imaginary reference line

- Road topography (3D) and topology
  
  - continuous geometry definition

see OpenDRIVE Format Specification, Rev. 1.5
Make OpenDRIVE data GIS-able

Geometry basics: types

```
| | speed
| |-planView
| | |-geometry
| | | |-line
| | | |-spiral
| | | |-arc
| | | |-poly3
| | | |-paramPoly3
| |-elevationProfile
```
Make OpenDRIVE data GIS-able
Geometry basics: discrete anchor points

```xml
<planView>
  <geometry s="0.0" x="604944.1037" y="5792860.1272" hdg="3.5148" length="9.7589">
    <arc curvature="9.0884E-4"/>
  </geometry>
  <geometry s="9.7589" x="604935.03" y="5792856.5285" hdg="3.5237" length="12.0">
    <line/>
  </geometry>
</planView>
```
Make OpenDRIVE data GIS-able
Geometry basics: continuous geometry evolution

```xml
<(planView>
  <geometry s="0.0" x="604944.1037"
    y="5792860.1272"
    hdg="3.5148"
    length="9.7589">
    <arc curvature="9.0884E-4"/>
  </geometry>
  <geometry s="9.7589" x="604935.03"
    y="5792856.5285"
    hdg="3.5237"
    length="12.0">
    <line/>
  </geometry>
</planView>
```
Make OpenDRIVE data GIS-able
Application-based discretisation (sampling)

“Everyone is doing this!”
Make OpenDRIVE data GIS-able
Application-based discretisation
Make OpenDRIVE data GIS-able
OGC Simple Feature primitives
Make OpenDRIVE data GIS-able

OGC Simple Feature primitives

see "Vector Data Models" in Geographic Information System Basics v1.0, CC BY-NC-SA 3.0
Make OpenDRIVE great again
OGC Simple Feature primitives

Point\((x, y)\)

LineString\((x_1, y_1, x_2, y_2, \ldots, x_n, y_n)\)
Make OpenDRIVE data GIS-able
Translation into Simple Feature model

```xml
<planView>
  <geometry s="0.0" x="604944.1037" y="5792860.1272"
            hdg="3.5148"
            length="9.7589">
    <arc curvature="9.0884E-4"/>
  </geometry>
  <geometry s="9.7589" x="604935.03" y="5792856.5285"
            hdg="3.5237"
            length="12.0">
    <line/>
  </geometry>
</planView>
```

LineString(
604944.1037 5792860.1272, 604752.81 5792819.10, ...)

LineString(
604935.03 5792856.5285, 604754.39 5792810.73, ...)
Make OpenDRIVE data GIS-able
Data binding to Java
Make OpenDRIVE data GIS-able
Geometry discretisation into Simple Feature model

OpenDRIVE

XML

custom data model with Simple Features
Solving “the problem” in three steps

1. Make OpenDRIVE data GIS-able
2. Deploy GIS data in spatial database
3. Publish as RESTful web service
Deploy GIS data in spatial database
Persisting through JPA with Hibernate Spatial

OpenDRIVE

custom data model
with Simple Features
Deploy GIS data in spatial database
Custom data model

<table>
<thead>
<tr>
<th>Simple Feature type</th>
<th>OpenDRIVE element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point</td>
<td>&lt;signal&gt;</td>
</tr>
<tr>
<td>LineString</td>
<td>driving &lt;lane&gt; boundary, &lt;roadMark&gt;, linear &lt;object&gt; (e.g. guardrail)</td>
</tr>
<tr>
<td>MultiLineString</td>
<td>road reference line &lt;planView&gt;</td>
</tr>
<tr>
<td>Polygon</td>
<td>driving &lt;lane&gt;, &lt;parkingSpace&gt;</td>
</tr>
</tbody>
</table>
Deploy GIS data in spatial database
Custom data model

<table>
<thead>
<tr>
<th>Simple Feature type</th>
<th>OpenDRIVE element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point</td>
<td>&lt;signal&gt;</td>
</tr>
<tr>
<td>LineString</td>
<td>driving &lt;lane&gt; boundary, &lt;roadMark&gt;, linear &lt;object&gt; (e.g. guardrail)</td>
</tr>
<tr>
<td>MultiLineString</td>
<td>road reference line &lt;planView&gt;</td>
</tr>
<tr>
<td>Polygon</td>
<td>driving &lt;lane&gt;, &lt;parkingSpace&gt;</td>
</tr>
</tbody>
</table>
Deploy GIS data in spatial database
Custom data model

Simple Feature type | OpenDRIVE element
--- | ---
Point | `<signal>`
LineString | `driving <lane> boundary, <roadMark>, linear <object> (e.g. guardrail)`
MultiLineString | `road reference line <planView>`
Polygon | `driving <lane>, <parkingSpace>`

Additional raw XML elements “attached”
- `<road>`
- `<junction>`
Solving “the problem” in three steps

1. Make OpenDRIVE data GIS-able
2. Deploy GIS data in spatial database
3. Publish as RESTful web service
Publish as RESTful web service
Directly from database through GeoServer

custom data model with Simple Features
Publish as RESTful web service
Publish as RESTful web service
The power of GeoServer

- Detailed **user management**
- Fine-grained **data security/access policies**
- OGC-standardised **REST API** (this “web thing”)
  - Web Map Service (WMS)
  - Web Feature Service (WFS)
  - Web Processing Service (WPS)
- Data output as image, KML, GML, GeoJSON, CSV, Shapefile, etc.
- Easy **snippet extraction** through custom **extension**
- Benefiting **spatial indices** on data → “fast like hell”
- Scalability
- Integration into most GIS tools
- Bla, bla, bla …
Live demo

OpenDRIVE through GeoServer
OpenDRIVE in QGIS
OpenDRIVE subset/snippet extraction queries
Conclusion: “Don’t re-invent the wheel”

• Geometry discretization should be based on OGC Simple Features

• Benefit from well-established tools in GIS domain:
  
  • Free/open frameworks for Java, C++, Python, … and web development
  
  • Super-easy ad hoc combination with arbitrary geo-data
  
  • Direct conversion into 100+ other formats: KML, GML, GeoJSON, CSV, Shapefile, SQLite, XLSX, …
  
  → GDAL: “One library to rule them all”
  
• Standardized web service interfaces already implemented (OGC WMS, WFS, …)

This ecosystem enables fast, large-scale serving of OpenDRIVE
https://youtu.be/diEnIUT6HmA