Project:	*OpenDRIVE®*		Document No.	Issue:
Title:	Format Specification, Rev. 1.4 - *DRAFT*		VI2014.106	E
	Date:	February 23, 2015	no. of pages:	102
	Issuing Party:	VIRES Simulationstechnologie GmbH		
	Author:	Marius Dupuis e.a.		
	Distribution List:	public		

issued by
VIRES Simulationstechnologie GmbH
Grassinger Strasse 8
83043 Bad Aibling
Germany
phone +49.8061.939093-0
fax +49.8061.939093-13
e-mail opendrive@vires.com
web www.vires.com

Note:
Modifications to the previous release are highlighted with yellow color.
(Draft E as of Feb. 16th was still using the original multi-color highlighting)
For new chapters / items, only the title of the respective chapter has been highlighted.

OpenDRIVE® Format Specification, Rev. 1.4 - DRAFT

VIRES Simulationstechnologie GmbH (VIRES) PROVIDES THIS MATERIAL AS IS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

VIRES may make improvements and changes to the format described in this manual at any time with no more notice than a statement on the OpenDRIVE website (www.opendrive.org). VIRES assumes no responsibility for the use of the product or this document except as expressly set forth in the applicable VIRES agreement or agreements and subject to terms and conditions set forth therein and applicable VIRES policies and procedures. This document may contain technical inaccuracies or typographical errors. Periodic changes may be made to the information contained herein. If necessary, these changes will be incorporated in new editions of the document.

VIRES is the owner of all intellectual property rights in and to this document and any proprietary software that accompanies this documentation, including but not limited to, copyrights in and to this document and any derivative works therefrom. Use of this document is subject to the terms and conditions of the VIRES Software License Statement included with this product.

Printed in Germany
February 2015

PUBLIC LICENSE STATEMENT
FOR
OpenDRIVE® Format Specification

OpenDRIVE is the property of VIRES Simulationstechnologie GmbH (VIRES) and is protected under the copyright and trademark laws of the Federal Republic of Germany. VIRES hereby grants to User a non-exclusive, non-transferable limited right to use OpenDRIVE as follows:

a. For reading OpenDRIVE into a computer program or database for in-house use, or as a feature of a commercial product.
b. For writing data from a computer program or database into OpenDRIVE for in-house use or as a feature of a commercial product.

Any attempt to sub-license, assign, or transfer all or any part of the OpenDRIVE specification is prohibited without the prior written consent of VIRES.

OpenDRIVE is a registered trademark of VIRE. User is not obliged but kindly asked to indicate that VIRES is owner of OpenDRIVE in any of User's published references to the trademark. User is not obliged but kindly asked to refer to www.opendrive.org in any of User’s published references to OpenDRIVE.

OPENDRIVE IS OFFERED FOR USE BY USER "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. ALL SUCH WARRANTIES ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL VIRES BE RESPONSIBLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF USE OF OPENDRIVE.

Editor's note:

This document is a revised draft version of the OpenDRIVE standard, release 1.4. It has gone through multiple critical reviews and is assumed to represent a mature state. Users are kindly asked to review the document and check for its applicability to their needs. Slight modifications may still be incorporated before the final release is due (no later than mid-April 2015).

If you have any questions or suggestions, please send an e-mail to

opendrive@opendrive.org

Editor's note: changes between 1.4, draft C and 1.4, draft D:

a) (railroad/bus) stations have been moved from road level to one level above roads
b) some attributes may be defined with an explicit "unit" identifier
c) changed the layout of the definitions for "nicer appearance"
d) re-named "Arguments" to "Attributes"
e) instead of using "Maximum Instances" and "Optional" for a bead, only "instances" is given and may have the following values:
   a. 1 exactly one instance, required
   b. 0..n maximum n instances (typically n=1), optional
   c. 0+ unlimited number of instances, optional
   d. 1+ minimum one instance, required
f) introduced "speed" bead below the road type definition, so that a default speed can be defined for a given road type at a given location
g) in chapter 5.3.4.1.5 (parametric cubic curves), it is proposed to allow custom definition of the parameter p's range
h) the "Lane Border Record" may also be defined for lane zero
i) the "Lane Speed Record" may have an optional attribute "unit"
j) added "unit" and "value" columns to attribute tables
k) added child beads for the object record for further definition of a parking space's appearance

Table of Contents

1 Preface

1.1 Scope

The OpenDRIVE® format provides a common base for describing track-based road networks using XML syntax. The data stored in an OpenDRIVE® file describes – in an analytical way – the geometry of roads as well as features along the roads that influence the logics (e.g. lanes, signs, signals).

The format is organized in nodes which can be extended with user-defined data. By this, a high degree of specialization for individual applications is feasible while maintaining the commonality required for the exchange of data between different applications.

1.2 Developers

The initial OpenDRIVE® format was developed by VIRES Simulationstechnologie GmbH, Germany, in close co-operation with Daimler Driving Simulator, Sindelfingen, Germany.

The contents of the file format are reviewed by a core team before publication. For the current members of the core team, please visit the OpenDRIVE® website (www.opendrive.org).

This standard has been created for its users. So, if you feel anything is missing, should be clarified or modified, please don't hesitate to contact us.

1.3 Point of Contact

Further assistance on OpenDRIVE® is provided

via the OpenDRIVE® website

http://www.opendrive.org

via email

opendrive@opendrive.org

and via the "classic style" of direct contact with human beings:

VIRES Simulationstechnologie GmbH

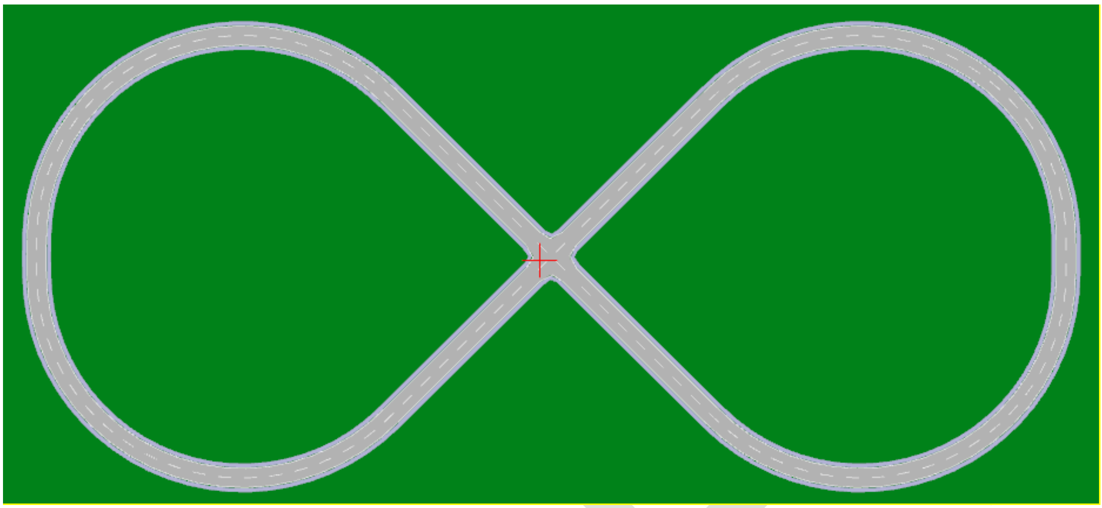
VIRES Simulationstechnologie GmbH
Grassinger Strasse 8
83043 Bad Aibling
Germany
phone +49.8061.939093-0
fax +49.8061.939093-13

A bug- and feature-tracking system has been introduced for OpenDRIVE which you may use for reporting and monitoring issues (i.e. tickets). For the former operation, a registration is mandatory. The system may be accessed via

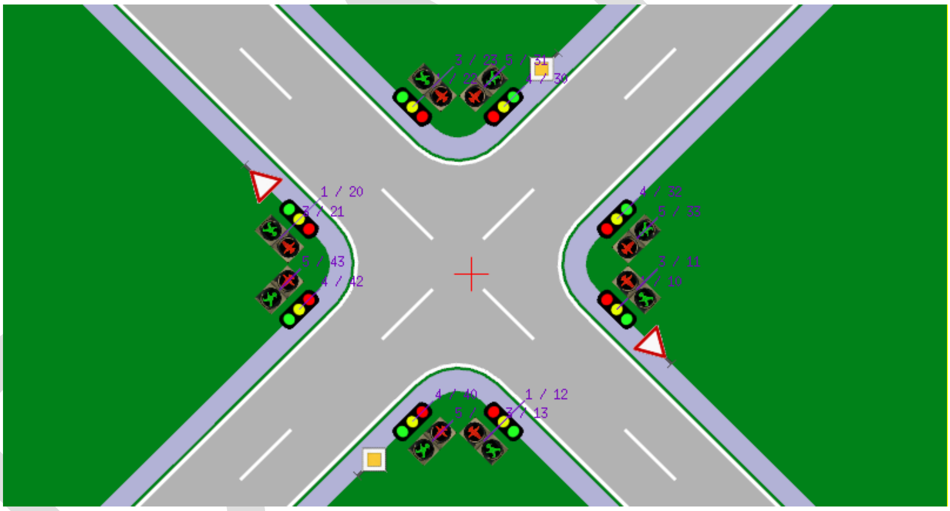
http://tracking.vires.com

1.4 Example for a Quick Start

The following example shall guide through this specification. It does not contain all available nodes of the format but illustrates very well the key principles. It's an endless two-lane country road with a four-way crossing in the center.



The example also contains traffic signs and traffic lights at the intersection:



The structure of the OpenDRIVE file of this small road network shall be explained in excerpts on the following pages. For the complete example file, please see the download section on the OpenDRIVE website.

All tags are enclosed by the OpenDRIVE declaration:

```
<?xml version="1.0" standalone="yes"?>
<OpenDRIVE>
:
</OpenDRIVE>
```

The header contains basic information about the road network:

```
    <header revMajor="1" revMinor="1" name="" version="1.00" date="Thu Dec 10 10:35:57 2009" north="0.0000000000000000e+00"
            south="0.0000000000000000e+00" east="0.0000000000000000e+00" west="0.0000000000000000e+00">
    </header>
```

Each road is described independently in type, plan view, elevation, cross section etc. with the respective link information to other roads in the same network:

```
    <road name="" length="1.6517824248160636e+01" id="500" junction="2">
        <link>
            <predecessor elementType="road" elementId="502" contactPoint="start" />
            <successor elementType="road" elementId="514" contactPoint="start" />
        </link>
        <type s="0.0000000000000000e+00" type="town"/>
        <planView>
            <geometry s="0.0000000000000000e+00" x="-7.0710678117841717e+00" y="7.0710678119660715e+00"
                      hdg="5.4977871437752235e+00" length="4.8660000002386400e-01">
                <line/>
            </geometry>
            <geometry s="4.8660000002386400e-01" x="-6.7269896520425938e+00" y="6.7269896522231525e+00"
                       hdg="5.4977871437736381e+00" length="3.1746031746031744e+00">
                <spiral curvStart="-0.0000000000000000e+00" curvEnd="-1.2698412698412698e-01"/>
            </geometry>
            <geometry s="3.6612031746270386e+00" x="-4.6416930098385274e+00" y="4.3409250448366459e+00"
                       hdg="5.2962250374496271e+00" length="9.1954178989066371e+00">
                <arc curvature="-1.2698412698412698e-01"/>
            </geometry>
            <geometry s="1.2856621073533674e+01" x="-4.6416930098799849e+00" y="-4.3409256447923106e+00"
                      hdg="4.1285529233027525e+00" length="3.1746031746031744e+00">
                 <spiral curvStart="-1.2698412698412698e-01" curvEnd="-0.0000000000000000e+00"/>
            </geometry>
            <geometry s="1.6031224248136848e+01" x="-6.7269896521209764e+00" y="-6.7269902521517775e+00"
                      hdg="3.9269908169787415e+00" length="4.8660000002378989e-01">
                <line/>
            </geometry>
        </planView>
        <elevationProfile>
            <elevation s="0.0000000000000000e+00" a="0.0000000000000000e+00" b="0.0000000000000000e+00"
                       c="0.0000000000000000e+00" d="0.0000000000000000e+00"/>
        </elevationProfile>
        <lanes>
            <laneSection s="0.0000000000000000e+00">
                <center>
                    <lane id="0" type="driving" level= "0">
                        <link>
                        </link>
                        <roadMark sOffset="0.0000000000000000e+00" type="none" weight="standard" color="standard"
                                  width="1.3000000000000000e-01"/>
                    </lane>
                </center>
                <right>
                    <lane id="-1" type="driving" level= "0">
                        <link>
                            <predecessor id="1"/>
                            <successor id="-1"/>
                        </link>
                        <width sOffset="0.0000000000000000e+00" a="3.7500000000000000e+00" b="0.0000000000000000e+00"
                               c="0.0000000000000000e+00" d="0.0000000000000000e+00"/>
                        <roadMark sOffset="0.0000000000000000e+00" type="solid" weight="standard" color="standard"
                                  width="1.3000000000000000e-01"/>
                    </lane>
                    <lane id="-2" type="border" level= "0">
                        <link>
                            <predecessor id="2"/>
                            <successor id="-2"/>
                        </link>
                        <width sOffset="0.0000000000000000e+00" a="3.4999999999999998e-01" b="0.0000000000000000e+00"
                               c="0.0000000000000000e+00" d="0.0000000000000000e+00"/>
                        <roadMark sOffset="0.0000000000000000e+00" type="none" weight="standard" color="standard"
                                  width="1.3000000000000000e-01"/>
                    </lane>
                    <lane id="-3" type="sidewalk" level= "0">
                        <link>
                            <predecessor id="3"/>
                            <successor id="-3"/>
                        </link>
                        <width sOffset="0.0000000000000000e+00" a="1.5000000000000000e+00" b="0.0000000000000000e+00"
                                        c="0.0000000000000000e+00" d="0.0000000000000000e+00"/>
                        <height sOffset="0.0000000000000000e+00" heightInner="1.2000000000000000e-01"
                                heightOuter="1.2000000000000000e-01"/>
                    </lane>
                    <lane id="-4" type="none" level= "0">
                        <link>
                        </link>
                        <width sOffset="0.0000000000000000e+00" a="2.0000000000000000e+00" b="0.0000000000000000e+00"
                               c="0.0000000000000000e+00" d="0.0000000000000000e+00"/>
                        <roadMark sOffset="0.0000000000000000e+00" type="none" weight="standard" color="standard"
                                  width="1.3000000000000000e-01"/>
                    </lane>
                </right>
            </laneSection>
        </lanes>
    </road>
```

Signals and signs are defined per road:

```
    <road name="" length="3.9999995757174296e+00" id="514" junction="-1">
        <link>
          :
        </link>
        <type s="0.0000000000000000e+00" type="town"/>
        <planView>
          :
        </planView>
        <elevationProfile>
          :
        </elevationProfile>
        <lateralProfile>
        </lateralProfile>
        <lanes>
          :
        </lanes>
        <objects>
        </objects>
        <signals>
            <signal s="1.0000000000000001e-01" t="5.5999999999999996e+00" id="40" name="_Sg40" dynamic="yes" orientation="-"
                    zOffset="0.0000000000000000e+00" type="1000001" country="OpenDRIVE" subtype="-1"
                    value="-1.0000000000000000e+00"/>
            <signal s="1.0000000000000001e-01" t="5.5999999999999996e+00" id="41" name="_Sg41" dynamic="yes" orientation="-"
                    zOffset="0.0000000000000000e+00" type="1000002" country="OpenDRIVE" subtype="-1"
                    value="-1.0000000000000000e+00"/>
            <signal s="1.0000000000000001e-01" t="-5.5999999999999996e+00" id="42" name="_Sg42" dynamic="yes" orientation="-"
                    zOffset="0.0000000000000000e+00" type="1000001" country="OpenDRIVE" subtype="-1"
                    value="-1.0000000000000000e+00"/>
            <signal s="1.0000000000000001e-01" t="-5.5999999999999996e+00" id="43" name="_Sg43" dynamic="yes" orientation="-"
                    zOffset="0.0000000000000000e+00" type="1000002" country="OpenDRIVE" subtype="-1"
                    value="-1.0000000000000000e+00"/>
            <signal s="3.8999999999999999e+00" t="5.5999999999999996e+00" id="0" name="" dynamic="no" orientation="-"
                    zOffset="0.0000000000000000e+00" type="306" country="OpenDRIVE" subtype="-1"
                    value="-1.0000000000000000e+00"/>
        </signals>
    </road>
```

Junctions help resolving otherwise ambiguous predecessor-successor relationships:

```
<OpenDRIVE>
          :
    <junction name="" id="2">
        <connection id="0" incomingRoad="502" connectingRoad="500" contactPoint="start">
            <laneLink from="1" to="-1"/>
            <laneLink from="2" to="-2"/>
            <laneLink from="3" to="-3"/>
        </connection>
        <connection id="1" incomingRoad="502" connectingRoad="510" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <connection id="2" incomingRoad="502" connectingRoad="506" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <connection id="3" incomingRoad="514" connectingRoad="511" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <connection id="4" incomingRoad="514" connectingRoad="515" contactPoint="start">
            <laneLink from="1" to="-1"/>
            <laneLink from="2" to="-2"/>
            <laneLink from="3" to="-3"/>
        </connection>
        <connection id="5" incomingRoad="514" connectingRoad="507" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <connection id="6" incomingRoad="516" connectingRoad="512" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <connection id="7" incomingRoad="516" connectingRoad="517" contactPoint="start">
            <laneLink from="1" to="-1"/>
            <laneLink from="2" to="-2"/>
            <laneLink from="3" to="-3"/>
        </connection>
        <connection id="8" incomingRoad="516" connectingRoad="505" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <connection id="9" incomingRoad="501" connectingRoad="513" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <connection id="10" incomingRoad="501" connectingRoad="503" contactPoint="start">
            <laneLink from="1" to="-1"/>
            <laneLink from="2" to="-2"/>
            <laneLink from="3" to="-3"/>
        </connection>
        <connection id="11" incomingRoad="501" connectingRoad="504" contactPoint="start">
            <laneLink from="1" to="-1"/>
        </connection>
        <controller id="1" type="0"/>
        <controller id="3" type="0"/>
        <controller id="4" type="0"/>
        <controller id="5" type="0"/>
    </junction>
</OpenDRIVE>
```

Finally, controllers provide a means for groupin dynamic signals:

```
<OpenDRIVE>
          :
    <controller name="ctrl001" id="1">
        <control signalId="10" type="0" />
        <control signalId="12" type="0" />
        <control signalId="20" type="0" />
        <control signalId="22" type="0" />
    </controller>
    <controller name="ctrl003" id="3">
        <control signalId="11" type="0" />
        <control signalId="13" type="0" />
        <control signalId="21" type="0" />
        <control signalId="23" type="0" />
    </controller>
    <controller name="ctrl004" id="4">
        <control signalId="40" type="0" />
        <control signalId="42" type="0" />
        <control signalId="30" type="0" />
        <control signalId="32" type="0" />
    </controller>
    <controller name="ctrl005" id="5">
        <control signalId="41" type="0" />
        <control signalId="43" type="0" />
        <control signalId="31" type="0" />
        <control signalId="33" type="0" />
    </controller>
          :
</OpenDRIVE>
```

2 Conventions

2.1 Naming Conventions

In this document, the following conventions apply:

`data types` are given according to IEEE standard
`track` signifies the reference line of a road

2.2 Units

All numeric values within this specification are in SI units (unless explicitly indicated otherwise), e.g.:

position/distance	in [m]
angles	in [rad]
time	in [s]
speed	in [m/s]

Geographic positions will be given in the unit defined by the spatial coordinate system (e.g. [deg] for WGS 84 – EPSG 4326). See sections 2.3.5 and 5.2.1.

Some data tags provide you with a means to explicitly state the unit of a given value. If the unit is not given or if there is no means to state the unit, then the SI units (see above) apply. Units that may be used in explicit assignments are:

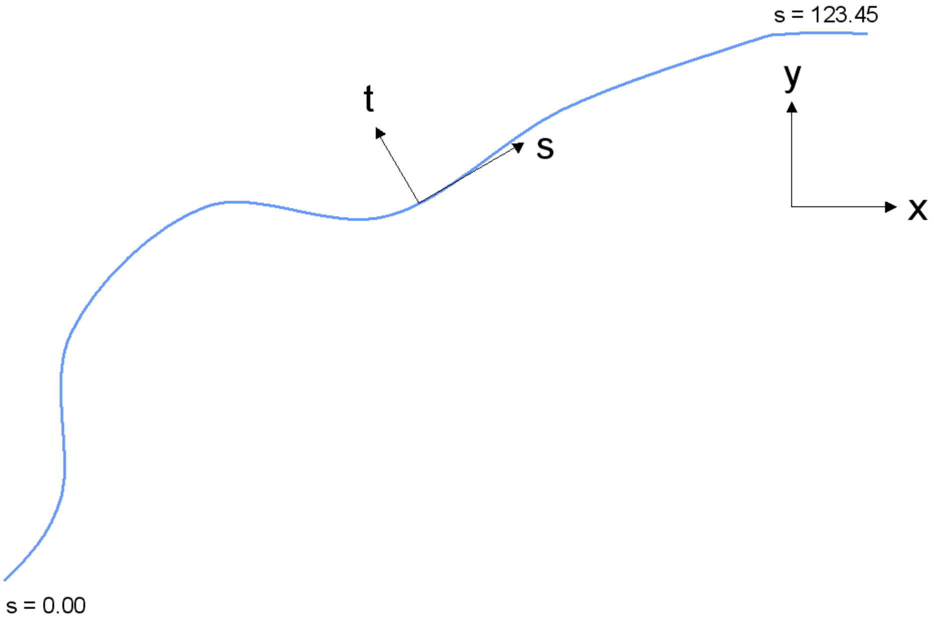
category	description	identifier
distance	meter	`m`
	kilometer	`km`
	feet	`ft`
	land mile	`mile`
speed	meters per second	`m/s`
	miles per hour	`mph`
	kilometers per hour	`km/h`
mass	kilogram	`kg`
	metric tons	`t`
slope	percent	`%`

Important note: these optional units shall only be used for purposes of signage and speed indication, NOT as general units for, e.g. the definition of road geometry etc.

2.3 Co-ordinate Systems

2.3.1 Overview

The following figure gives an overview of the two co-ordinate systems most frequently used in this specification – track co-ordinates and inertial co-ordinates (for details, see the following chapters).



2.3.2 *Inertial System*

The inertial system is a right-handed co-ordinate system according to ISO 8855 with the axes pointing to the following directions:

x	forward
y	left
z	up

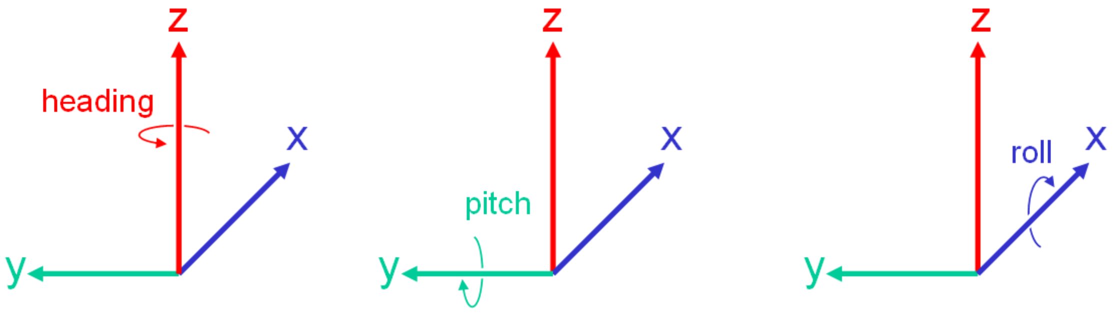
For geographic reference, the following convention applies:

x	east
y	north
z	up

Within the inertial system, the following angles are defined:

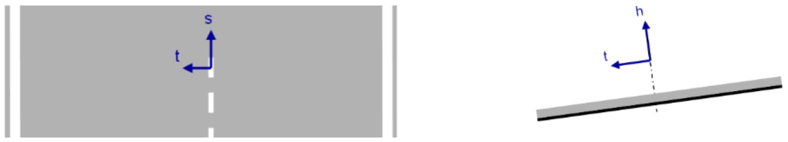
heading	around z-axis, 0.0 = east
pitch	around y-axis, 0.0 = level
roll	around x-axis, 0.0 = level

The following image shows the positive axes and positive directions of the corresponding angles.



2.3.3 Track System

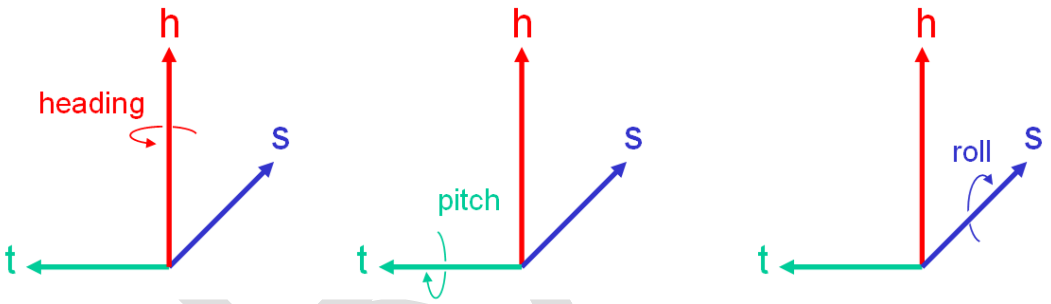
The track co-ordinate system applies along the reference line of a road. It is a right-handed co-ordinate system. The following degrees of freedom are defined:



s position along reference line, measured in [m] from the beginning of the track, calculated in the xy-plane (i.e. not taking into account the elevation profile of the track)
t lateral position, positive to the left
h up

heading around h-axis, 0 = forward
pitch around t-axis, 0 = level
roll around s-axis, 0 = level

The following image shows the positive axes and positive directions of the corresponding angles.



2.3.4 Local System

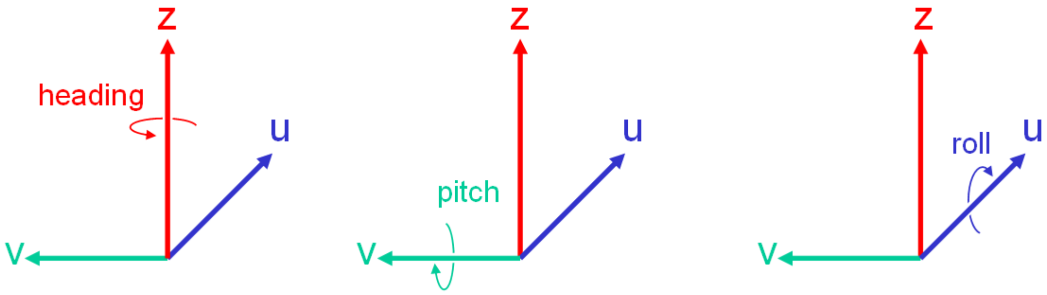
The local system is a right-handed co-ordinate system according to ISO 8855 with the axes pointing to the following directions:

u	forward
v	left
z	up

Within the local system, the following angles are defined:

heading	around z-axis, 0.0 = east
pitch	around v-axis, 0.0 = level
roll	around u-axis, 0.0 = level

The following image shows the positive axes and positive directions of the corresponding angles.



The local system can only be positioned in track space by providing the full track co-ordinates and orientation of its origin.

2.3.5 Geo-Referencing

Starting with OpenDRIVE 1.4, road networks may be geo-referenced using a projection definition via OGCWKT string.

2.3.6 Curvature

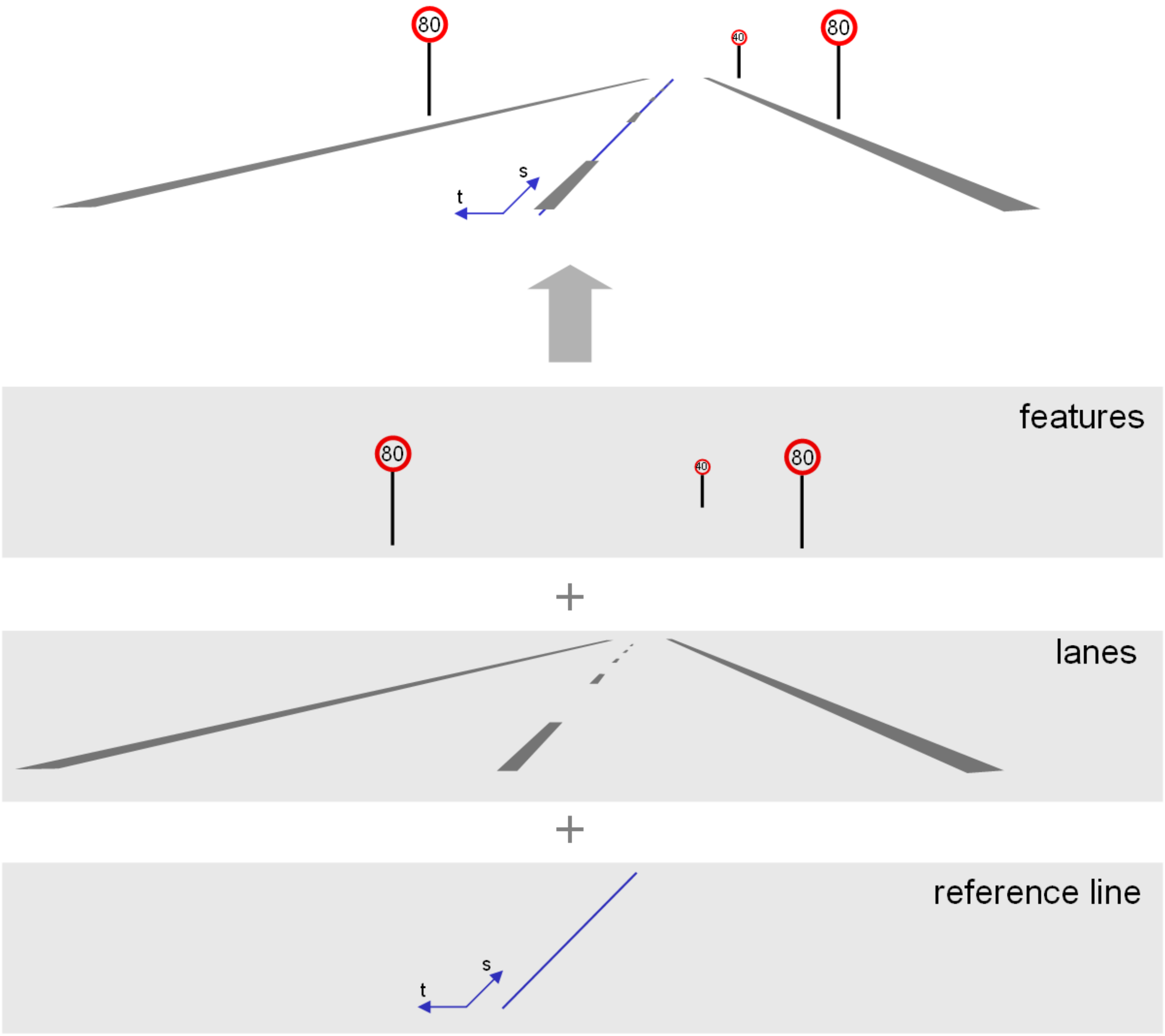
For curvature indications, the following convention applies:

- Positive curvature left curve (counter-clockwise motion)
- Negative curvature right curve (clockwise motion)

3 Road Layout

3.1 General

The following figure depicts the principles of road layout covered by this specification:



All roads consist of a reference line which defines the basic geometry (arcs, straight lines etc.). Along the reference line, various properties of the road can be defined. These are, e.g. elevation profile, lanes, traffic signs etc. Roads can be linked to each other either directly (when there is only one connection possible between two given roads) or via junctions (when more than one connection is possible from a given road to other roads).

All properties may be parameterized according to the standards laid out in this specification and, optionally, by user-defined data.

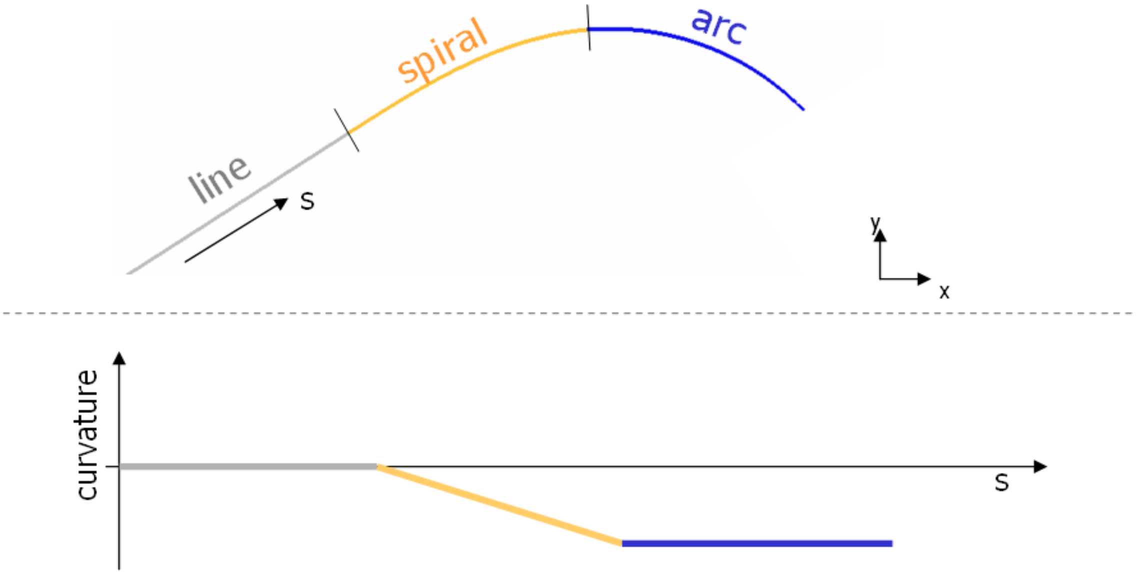
The convention applies that properties of the same type defined along a single reference line must be listed in ascending order. This means that the start co-ordinate (parameter s, see above) of a property must either be the same or greater than the start co-ordinate of the preceding property of same type on the same track.

3.2 Reference Line (Track)

The geometry of the reference line is described as a sequence of primitives of various types. The available primitives are:

- straight line (constant zero curvature)
- spiral (linear change of curvature)
- curve (constant non-zero curvature along run-length)
- cubic polynom
- parametric cubic curves

The following figure illustrates the composition of a reference line from some of the above mentioned elements.



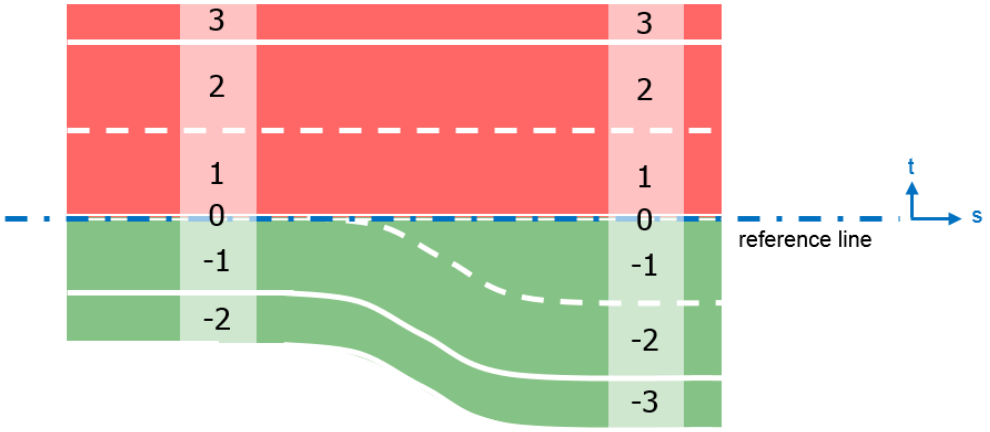
3.3 Lanes

3.3.1 *General*

Lanes are identified by numbers which are

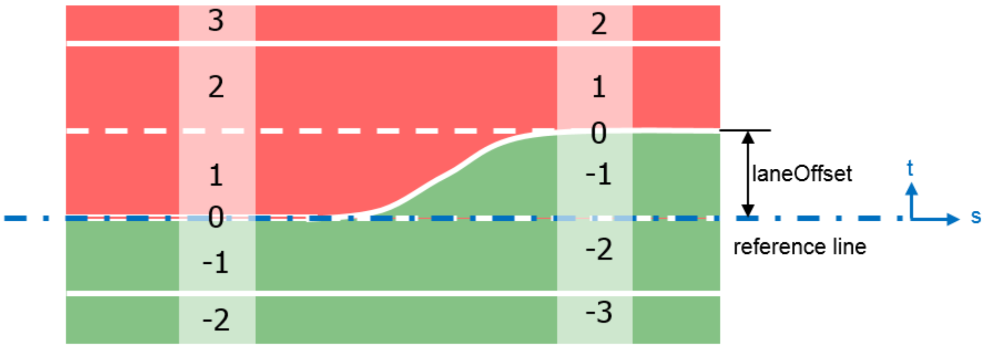
- unique (per lane section, see below)
- in sequence (i.e. without gaps),
- starting from 0 on the reference line
- ascending to the left (positive t-direction)
- descending to the right (negative t-direction)

The total number of lanes is not limited. The reference line itself is defined as lane zero and must not have a width entry (i.e. its width must always be 0.0).



3.3.2 *Lane Offset*

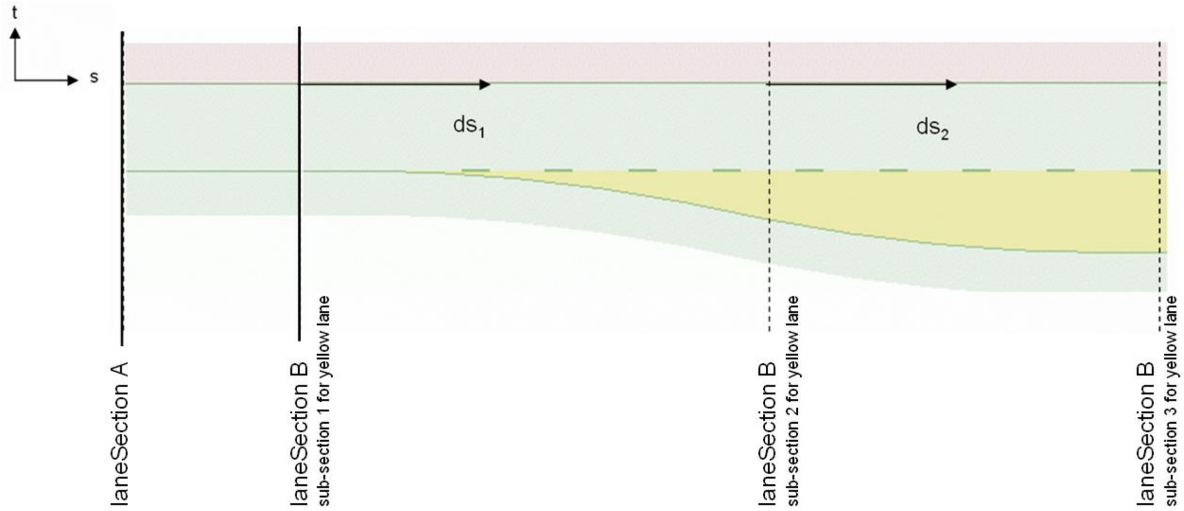
In certain situations (e.g. a left-turn lane inserted as inner-most lane) it may be more convenient to shift the lane profile instead of shifting the reference line. For this purpose, lane 0 may be offset using a cubic polynom:



3.3.3 Lane Sections

The lanes appearing in a given cross-section along the road are defined in so-called lane sections. Multiple lane sections may be defined in ascending order along a reference line. Each lane section is valid until the next lane section is defined. Therefore, in order to be usable, each road must at least be equipped with one lane section starting at s= 0.0m.

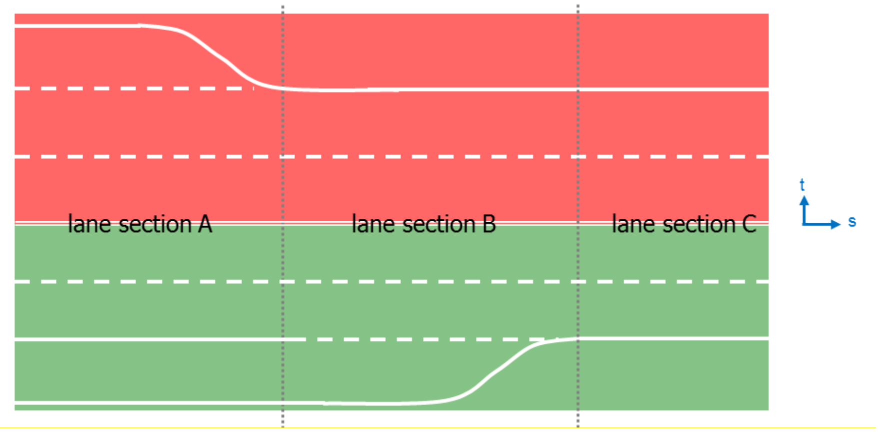
The following figure depicts the principles of lane sections:


t
s
ds_1
ds_2
laneSection A
laneSection B
sub-section 1 for yellow lane
laneSection B
sub-section 2 for yellow lane
laneSection B
sub-section 3 for yellow lane

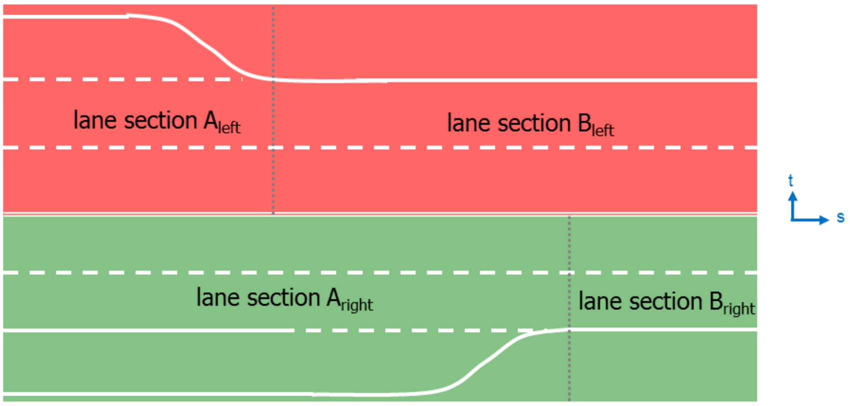
Per lane section, the number of lanes is constant. However, the properties of each lane (e.g. width, road marks, friction etc.) may change.

Until OpenDRIVE 1.4 it was mandatory that lane sections be defined for both directions of a road at identical locations. In certain cases (e.g. where an addition or subtraction of lanes happens on one side of a road only or where the number of lanes changes on both sides but with a slight longitudinal offset), this could lead to restrictions or increased effort for the definition of complex road layouts.

The following figure illustrates the classical definition of a typical motorway entry/exit with longitudinal offset of the respective entry / exit lanes:



With lane sections which are valid for one side only, the complexity of the definition can be reduced considerably:



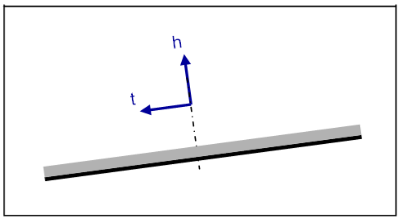
3.3.4 Lane Properties

Lane properties are defined relative to the start of the corresponding lane section. Offsets start at 0.0 for the beginning of the lane section and increase corresponding to the road co-ordinate s. Lane properties are valid until a new property of the same type is defined or the lane section ends. Lane properties of identical types must be defined in ascending order. Lane properties may be point or range properties, i.e. they are valid at a given point only or over a given range, respectively.

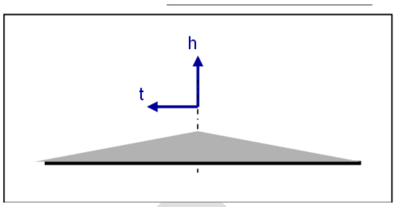
If a property is not defined within a given lane section or not covering the entire section, the application may assume default values.

3.4 Superelevation and Crossfall

In many cases, a road's cross section will not be parallel to the underlying terrain. Instead, it will be elevated to one side (e.g. in curves) or to the center (for drainage). Both properties are covered by the OpenDRIVE format with the former being called "*superelevation*" and the latter "*crossfall*". The following figure illustrates both properties:



superelevation

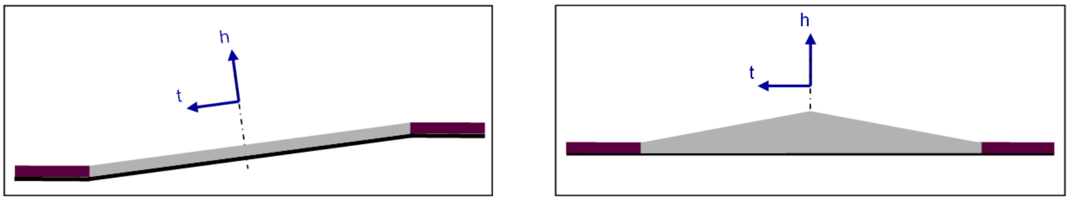


crossfall

Superelevation and *crossfall* can be superimposed in order to provide smooth transitions between e.g. straight sections with *crossfall* and curves with *superelevation*.

As can be seen from the above figures, the *superelevation* is defined per entire road cross section whereas the *crossfall* is defined per side of the road.

Single lanes can be excluded from the application of the *superelevation* and *crossfall* properties. Pedestrian walkways, for example, will always run on level planes (see following figure)



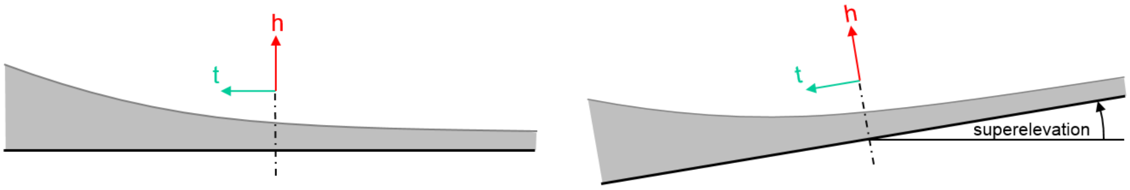
The definition of *superelevation* entries will not alter the actual width of a lane but its projected width whereas for a *crossfall* definition, the behavior is vice versa.

With the introduction of the "*lateral profile*" (see next chapter), the *crossfall* property might be treated as a special case or become completely obsolete. It is recommended to prefer the definition of a *lateral profile* instead of the *crossfall* property in new databases.

3.5 Lateral Profile

Some lateral road shapes may be more complex than they can be described by superelevation and crossfall alone. One way of describing these complex shapes is the use of explicit surface data (see below). An analytical method for describing lateral profiles is added to the format specification with OpenDRIVE 1.4. This allows, for example, for the description of curved road surfaces as they can be found on high-speed test tracks. Note that the "superelevation" propery acts on the reference plane of the road, not on the lateral profile itself.

The lateral profile must not be combined with the "crossfall" property.



3.6 Road Linkage

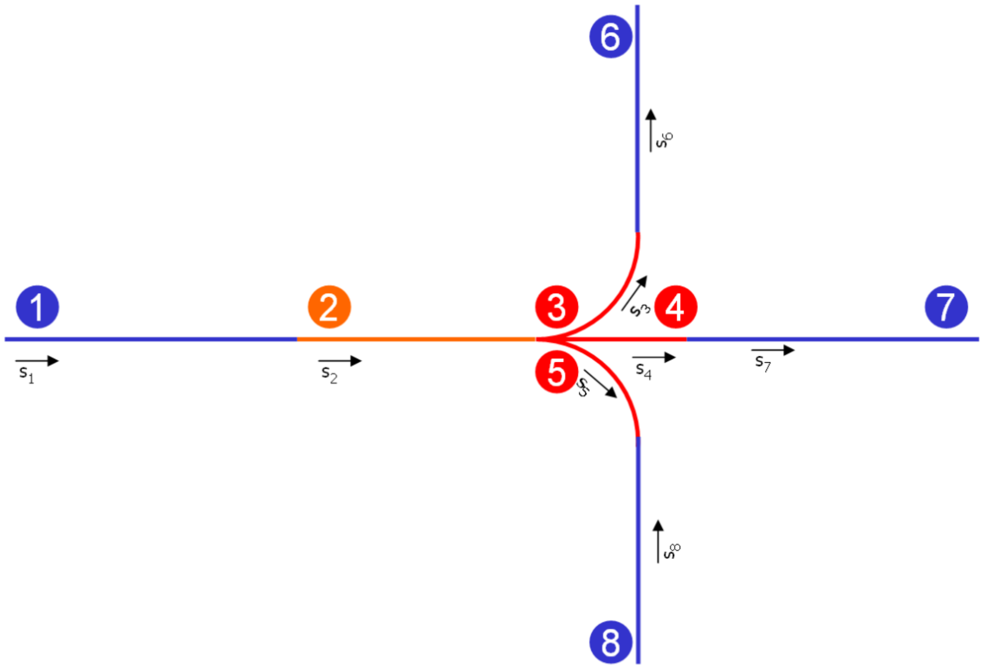
3.6.1 Overview

In order to navigate through a road network, the application must know about the linkage between different roads. Two types of linkage are possible:

- successor/predecessor linkage (*standard linkage*)
- junctions

Whenever the linkage between two roads is clear, a standard linkage information is sufficient. A junction is required when the successor or predecessor relationship of a road is ambiguous. Here, the application needs to select one of several possibilities.

The following figure and table illustrate the different cases:



Road	Predecessor	Successor
1	-	2
2	1	ambiguous
3	2	6
4	2	7
5	2	8
6	3	-
7	4	-
8	-	5

Clarification: The above table and figure illustrate some of the possible connections within a junction. The roads no. 3, 4 and 5 are called "connecting roads". For them, road no. 2 acts as the so-called “incoming road”. Roads 6, 7 and 8 are “outgoing roads” (they may also act as “incoming” roads for other connections not shown in the figure).

In order to facilitate navigation through a road network on a per-lane basis, additional linkage information can be provided on the lane level.

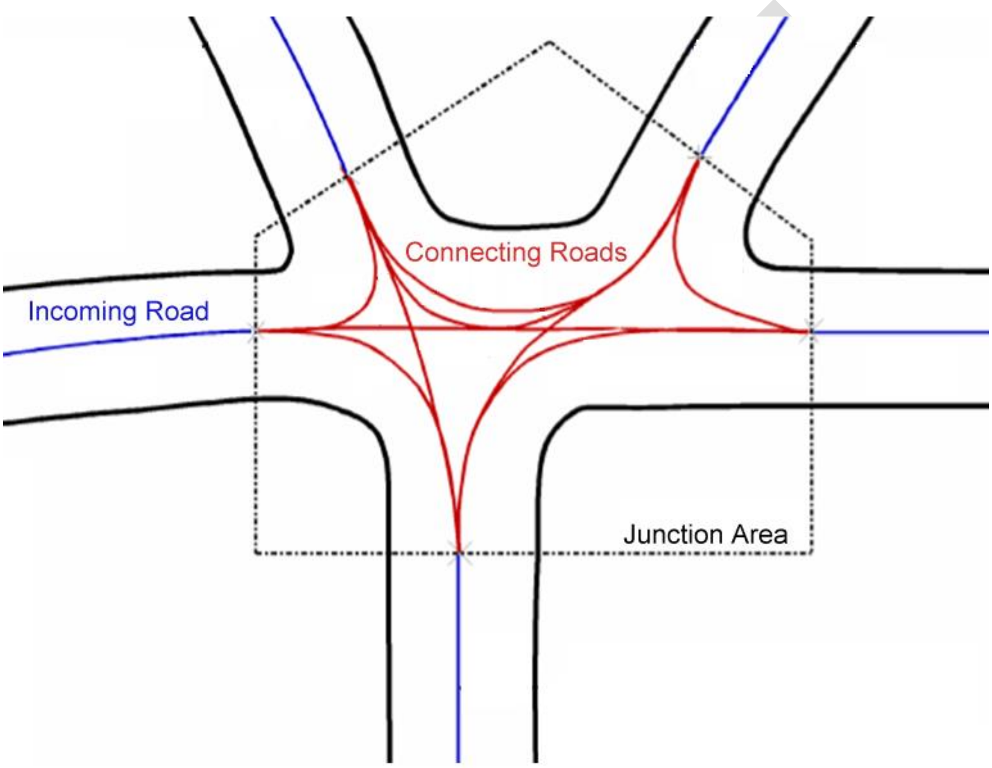
3.6.2 *Junctions*

3.6.2.1 Basics

The basic principle is very simple:

> Junctions link *in-coming roads* via paths (*connecting roads*) to *out-going roads*.

The following figure shows a complex junction scenario:



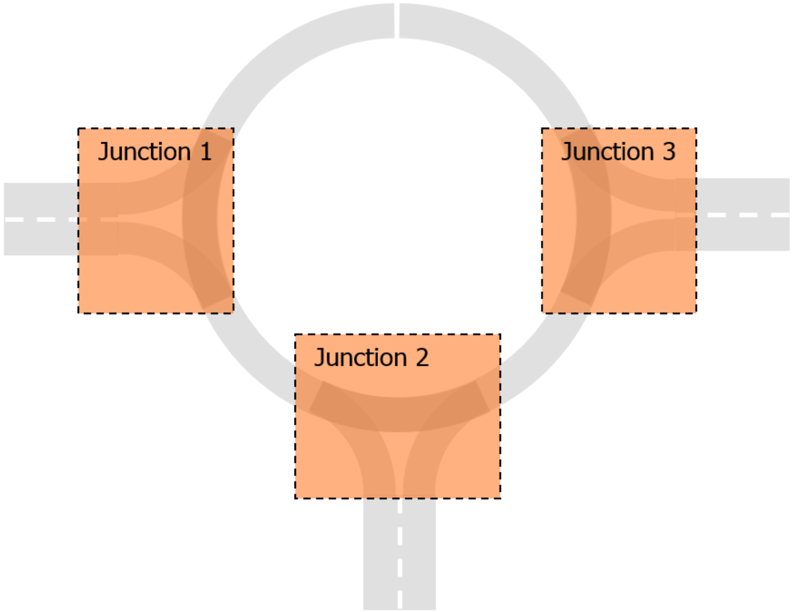
Connecting Roads are also modeled as roads according to the rules laid out here for all other roads. They consist of reference lines with lane sections etc. Usually, in-coming roads also serve as out-going roads, with the actual usage of a road being determined by its lanes.

Junctions consist of a connection matrix which indicates all possibilities to enter a connecting road from a given in-coming road. These connections are listed on a per-lane basis in order to facilitate navigation. Once a connecting road is entered, the following connection to the corresponding out-going road can be retrieved from the general successor/predecessor information that is stored with each road. Within the junctions, priorities of roads relative to each other may be stored but they may also be retrieved by evaluating the signs/signals and geometry.

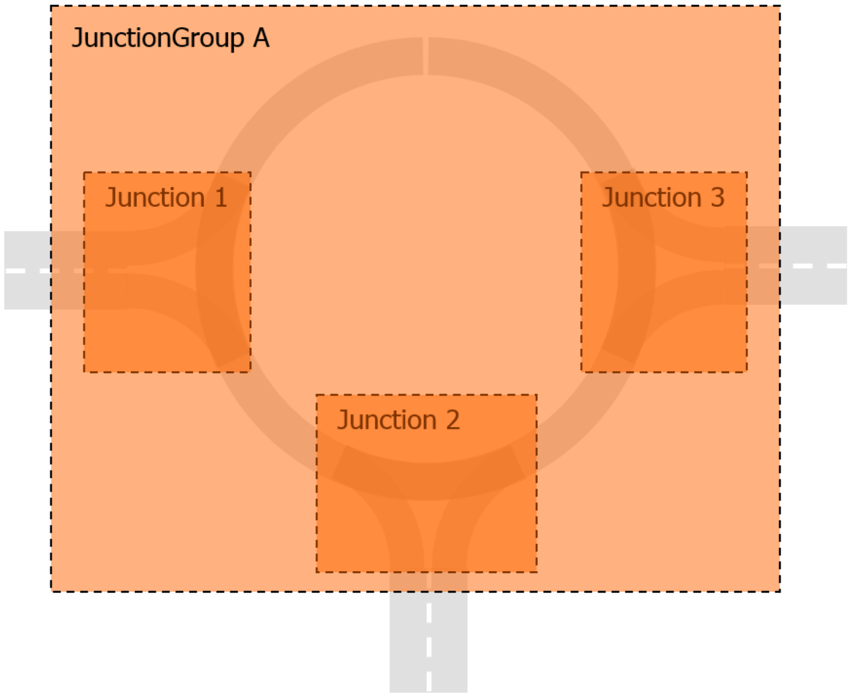
U-turns as part of a junction must be modeled as individual connecting roads, thus adding one more possibility to the connection matrix.

3.6.2.2 Junction Groups

In certain situations (e.g. roundabouts), junctions have to know about the existence of each other, so that traffic rules (right-of-way) can be implemented more efficiently. According to the OpenDRIVE style guide, a roundabout shall be composed of individual junctions:



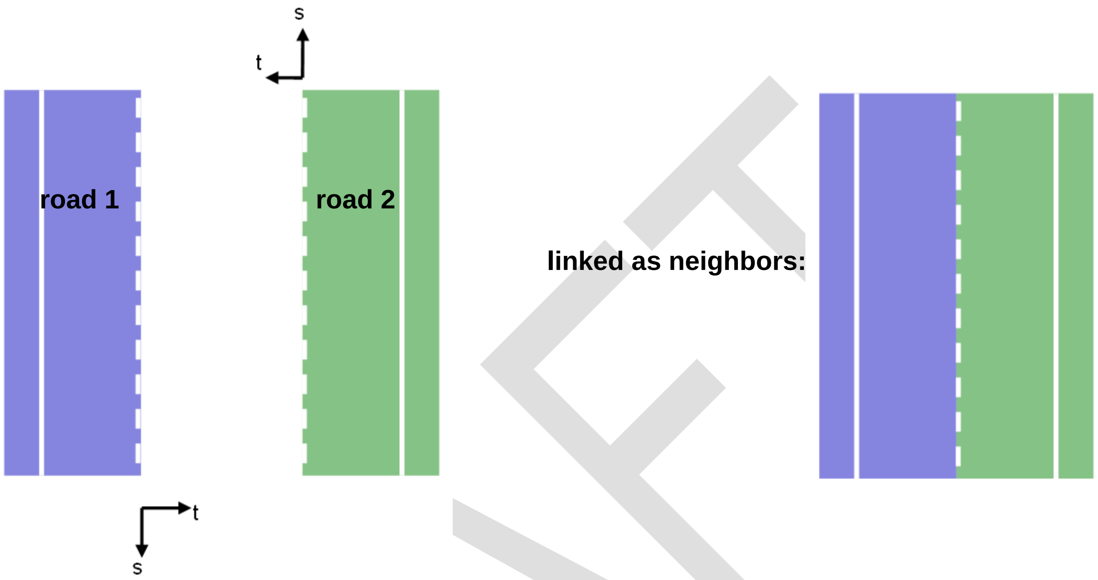
Putting all of these junctions into a single group can help the traffic simulation software considerably:



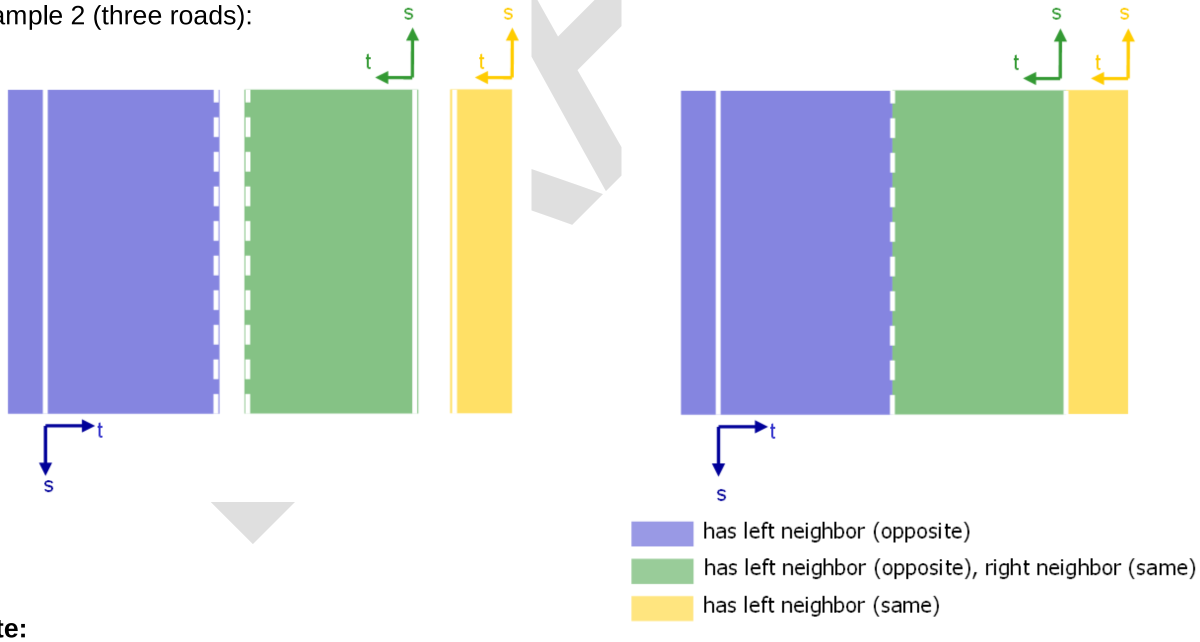
3.6.3 Neighbors

Roads may not only be linked to predecessors and successors but also to neighbors. This type of link information may be required when only one driving direction is defined per reference line (i.e. only left or only right lanes), or when only a fraction of the total number of lanes is defined per reference line. Each road may have up to two neighbors.

Example 1 (two roads being each other's neighbor):



Example 2 (three roads):


s
s
t
t
s
s
t
t
t
s
t
s
has left neighbor (opposite)
has left neighbor (opposite), right neighbor (same)
has left neighbor (same)

Note:
The neighbor entry has been introduced mainly for legacy purposes. For the design of new road networks, it is recommended to define both driving directions of a road along a single reference line and to avoid using the neighbor entry (this recommendation applies outside junctions only; within junctions, single reference lines may and should be used for each driving direction, however).

3.7 Surface

OpenDRIVE provides two approaches for describing surface properties:

- Standard description:

  In the standard case, a `<material>` record may be defined per lane of the road, providing parameters for
  - surface material code
  - roughness
  - friction

- Extended description:

  A more detailed description of road surface data (e.g. from measurements), which is not limited to the definition of material properties within lane boundaries, may be provided within the newly introduced `<surface>` record. This data may be applied to an entire cross section or parts thereof.

  Due to the potentially very large amount of data that is contained in detailed surface information and due to the fact that publicly available data formats already exist for surface data, OpenDRIVE does not define its own XML implementation of surface descriptions but provides references to the respective data files instead.

  Formats officially supported as surface import formats to OpenDRIVE are listed below. This list may be extended in future revisions of OpenDRIVE depending on the purposes served by additional formats.

  In order to guarantee the portability of an OpenDRIVE description, this standard recommends using the OpenCRG format as preferred extended road surface description.

  List of supported surface data formats:

no.	name	revision	description
1	Open Curved Regular Grid (OpenCRG)	June 2010	data format by TÜV Süd in co-operation with Daimler AG. Tools available by Daimler and VIRES Detailed information: http://www.opencrg.org

For the application of this data to the OpenDRIVE road see chapter 5.3.10.

3.8 Alternative Layouts

Depending on the individual application, it might be necessary to describe various possible setups of a road network’s properties, so that the OpenDRIVE file does not only contain one description of these properties but provides access to all pre-defined setups.

For this purpose, the `<set>` record is being introduced (see 5.10). It allows the user to define alternative instances of a property within its level.

The `<set>` record may be used at any level without restrictions, however users should take into account that the portability of an OpenDRIVE road description may suffer with sets being defined at levels where another user might not expect them. In the future, this document shall contain hints where sets have been used successfully in applications, in order to give an indication where sets are encouraged and where they should be avoided.

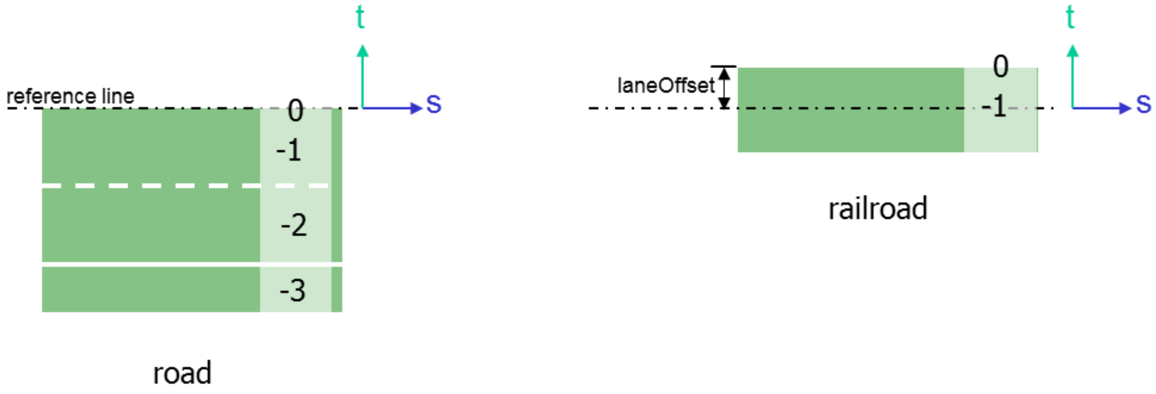
3.9 Railroad Elements

DISCLAIMER: *OpenDRIVE has not been designed to describe complex railroad networks including signaling, safety systems (e.g. ERTMS) etc. For this purpose, other formats exist (e.g. http://www.railml.org). Just in cases where rails and roads meet, shall OpenDRIVE provide a means to describe both features in a single file and format.*

Road networks in urban areas sometimes have to include railroad elements. This is especially true for trams (i.e. streetcars) which either share space with other vehicles or are operated in designated areas. In many cases, shared / exclusive space usage varies continuously within a network.

3.9.1 Railroad Tracks

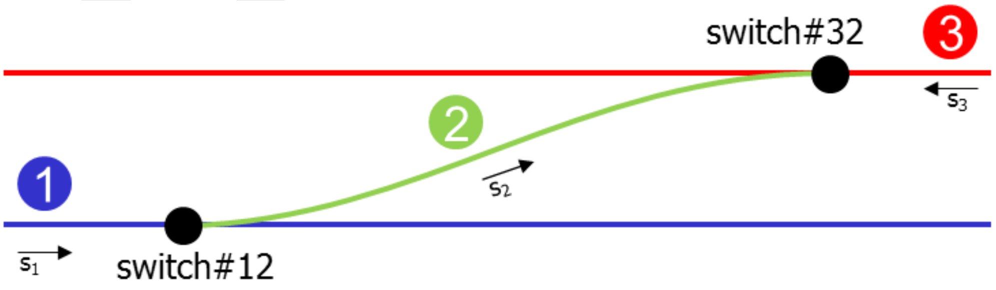
For rail-based vehicles, the freedom of lateral motion is quite restricted (except for the case of de-railing). Therefore, it is sufficient to provide one lane for a railroad which is of type "tram" or – more general – "rail" and which is centered around the reference line (this can be achieved by applying an offset to lane 0, see figure):



NOTE: For railroad lines with multiple parallel tracks (typically dual-track layout), it is mandatory to use one discrete track per direction instead of adding railroad lanes to an existing track. It is also recommended to provide individual reference lines (each bearing exactly one lane) for rail-based vehicles even in areas of shared use with road vehicles.

3.9.2 Railroad Switches

Where junctions for road vehicles can get quite complex, switches in rails always provide a clear path depending on their position. Therefore, no connection matrix is required and switches may be placed even within tracks (i.e. the main track's reference line may continue), see the following figure:

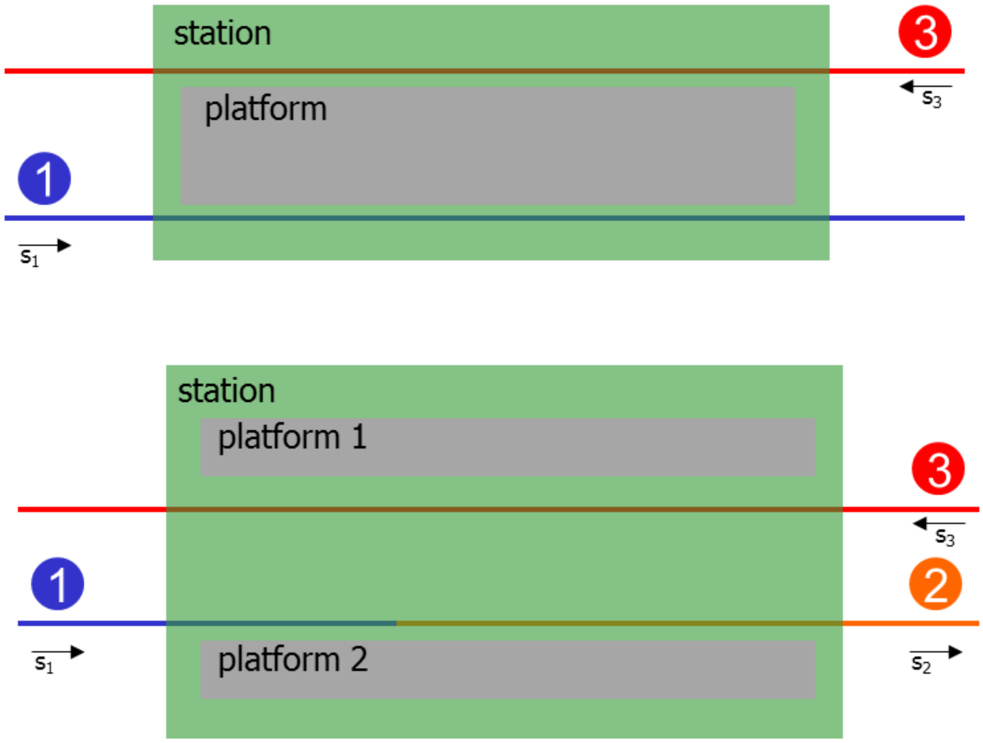


Tracks 1 and 3 are the main tracks whereas track 2 links track 1 and 3. The path driven by a rail-based vehicle only depends on the logical states of switches #12 and #32. Therefore, directional control is executed by the network on the vehicle (quite the opposite of what can be done with road-based vehicles).

3.9.3 Railroad Stations

Railroad lines may contain stations and these may contain platforms. In order to define the spatial extent of a station, at least one platform entry is required.

Platform IDs have to be unique but several tracks may reference the same platform. See also the following figure. In the situation on top, tracks 1 and 3 will have a reference to the same platform, in the bottom situation, tracks 1 and 2 will refer to platform 2 whereas track 3 will refer to platform 1 only.



4 File

4.1 Format

OpenDRIVE® data is stored in an XML file.

4.2 Extension

OpenDRIVE® files have the extension ".xodr".
Compressed OpenDRIVE® files have the extension ".xodrz".

4.3 Structure

The OpenDRIVE® file structure is laid out according to XML rules and with reference to the respective schema file.

Beads are organized in levels. Beads with a level greater than zero (0) are children of the preceding level. Beads with a level of one (1) are called primary beads.

Each bead can be extended with user-defined data. This data is stored in so-called ancillary beads.

4.4 Notation

All floating point numbers are "double" precision by default. It is highly recommended to use a 16 digit scientific representation for floating point numbers.

4.5 Schema

The schema file for the OpenDRIVE® format can be retrieved from *www.opendrive.org*

4.6 Combining Files

Multiple files can be combined by means of an `include` tag at the appropriate locations. Upon parsing this tag, OpenDRIVE readers shall immediately start reading the file specified as attribute of the tag. It is the user's responsibility to make sure that contents read from an include file are consistent with the context from which the inclusion starts.

4.7 Overview of Beads

The following table provides a simplified overview of all beads that may occur within an OpenDRIVE® file. It also indicates the maximum number of instances of a given bead type below a single parent. The naming convention is as follows:

- 1 exactly one instance, required
- 0..n maximum n instances (typically n=1), optional
- 0+ unlimited number of instances, optional
- 1+ minimum one instance, required

Children of optional beads must not be present if their respective parent is omitted. The levels of the beads are indicated by indentation and by the appropriate numbers.

bead name	level	instances per parent	parent	chapter
OpenDRIVE	0	1	-	5.1
|-header	1	1	OpenDRIVE	5.2
| |-geoReference	2	0..1	header	5.2.1
|-road	1	1+	OpenDRIVE	5.3.1
| |-link	2	0..1	road	5.3.2
| | |-predecessor	3	0..1	link	5.3.2.1
| | |-successor	3	0..1	link	5.3.2.2
| | |-neighbor	3	0..2	link	5.3.2.3
| |-type	2	0+	road	5.3.3
| | |-speed	3	0+	type	5.3.3.1
| |-planView	2	1	road	5.3.4
| | |-geometry	3	1+	planview	5.3.4.1
| | | |-line	4	1	geometry	5.3.4.1.1
| | | |-spiral	4	1	geometry	5.3.4.1.2
| | | |-arc	4	1	geometry	5.3.4.1.3
| | | |-poly3	4	1	geometry	5.3.4.1.4
| | | |-paramPoly3	4	1	geometry	5.3.4.1.5
| |-elevationProfile	2	0..1	road	5.3.5
| | |-elevation	3	0+	elevationProfile	5.3.5.1
| |-lateralProfile	2	0..1	road	5.3.6
| | |-superelevation	3	0+	lateralProfile	5.3.6.1
| | |-crossfall	3	0+	lateralProfile	5.3.6.2
| | |-shape	3	0+	lateralProfile	5.3.6.3
| |-lanes	2	1	road	5.3.7
| | |-laneOffset	3	0+	lanes	5.3.7.1
| | |-laneSection	3	1+	lanes	5.3.7.2
| | | |-left	4	0..1	lane section	5.3.7.2.1
| | | | |-lane	5	1+	left	5.3.7.2.1.1
| | | | | |-link	6	0..1	lane	5.3.7.2.1.1.1
| | | | | | |-predecessor	7	0..1	link	5.3.7.2.1.1.1.1
| | | | | | |-successor	7	0..1	link	5.3.7.2.1.1.1.2
| | | | | |-width	6	1+	lane	5.3.7.2.1.1.2
| | | | | |-border	6	0+	lane	5.3.7.2.1.1.3
| | | | | |-roadMark	6	0+	lane	5.3.7.2.1.1.4
| | | | | | | -type	7	0..1	roadMark	5.3.7.2.1.1.4.1
| | | | | | | | -line	8	1+	type	5.3.7.2.1.1.4.1.1
| | | | | |-material	6	0+	lane	5.3.7.2.1.1.5
| | | | | |-visibility	6	0+	lane	5.3.7.2.1.1.6
| | | | | |-speed	6	0+	lane	5.3.7.2.1.1.7
| | | | | |-access	6	0+	lane	5.3.7.2.1.1.8
| | | | | |-height	6	0+	lane	5.3.7.2.1.1.9
| | | |-center	4	0..1	lane section	5.3.7.2.1
| | | | |-lane	5	1	center	5.3.7.2.1.1
| | | | | |-link	6	0..1	lane	5.3.7.2.1.1.1
| | | | | | |-predecessor	7	0..1	link	5.3.7.2.1.1.1.1
| | | | | | |-successor	7	0..1	link	5.3.7.2.1.1.1.2
| | | | | |-roadMark	6	0+	lane	5.3.7.2.1.1.4
| | | | | | | -type	7	0..1	roadMark	5.3.7.2.1.1.4.1
| | | | | | | | -line	8	1+	type	5.3.7.2.1.1.4.1.1

bead name	level	instances per parent	parent	chapter
| | | |-right	4	0..1	lane section	5.3.7.2.1
| | | | |-lane	5	1+	right	5.3.7.2.1.1
| | | | | |-link	6	0..1	lane	5.3.7.2.1.1.1
| | | | | | |-predecessor	7	0..1	link	5.3.7.2.1.1.1.1
| | | | | | |-successor	7	0..1	link	5.3.7.2.1.1.1.2
| | | | | |-width	6	1+	lane	5.3.7.2.1.1.2
| | | | | |-border	6	0+	lane	5.3.7.2.1.1.3
| | | | | |-roadMark	6	0+	lane	5.3.7.2.1.1.4
| | | | | | | -type	7	0..1	roadMark	5.3.7.2.1.1.4.1
| | | | | | | | -line	8	1+	type	5.3.7.2.1.1.4.1.1
| | | | | |-material	6	0+	lane	5.3.7.2.1.1.5
| | | | | |-visibility	6	0+	lane	5.3.7.2.1.1.6
| | | | | |-speed	6	0+	lane	5.3.7.2.1.1.7
| | | | | |-access	6	0+	lane	5.3.7.2.1.1.8
| | | | | |-height	6	0+	lane	5.3.7.2.1.1.9
| |-objects	2	0..1	road	5.3.8
| | |-object	3	0+	objects	5.3.8.1
| | | |-repeat	4	0..1	object	5.3.8.1.1
| | | |-outline	4	0..1	object	5.3.8.1.2
| | | | |-cornerRoad	5	0+	outline	5.3.8.1.2.1
| | | | |-cornerLocal	5	0+	outline	5.3.8.1.2.2
| | | |-material	4	0..1	object	5.3.8.1.3
| | | |-validity	4	0+	object	5.3.8.1.4
| | | |-parkingSpace	4	0..1	object	5.3.8.1.5
| | | | |-marking	5	0..4	parkingSpace	5.3.8.1.6
| | |-objectReference	3	0+	objects	5.3.8.2
| | | |-validity	4	0+	objectReference	5.3.8.2.1
| | |-tunnel	3	0+	objects	5.3.8.3
| | | |-validity	4	0+	tunnel	5.3.8.3.1
| | |-bridge	3	0+	objects	5.3.8.4
| | | |-validity	4	0+	bridge	5.3.8.4.1
| |-signals	2	0..1	road	5.3.9
| | |-signal	3	1+	signals	5.3.9.1
| | | |-validity	4	0+	signal	5.3.9.1.1
| | | |-dependency	4	0+	signal	5.3.9.1.2
| | |-signalReference	3	0+	signals	5.3.9.2
| | | |-validity	4	0+	signalReference	5.3.9.2.1
| |-surface	2	0..1	road	5.3.10
| | |-CRG	3	0+	surface	5.3.10.1
| |-railroad	2	0..1	road	5.3.11
| | |-switch	3	0+	railroad	5.3.11.1
| | | |-mainTrack	4	1	switch	5.3.11.1.1
| | | |-sideTrack	4	1	switch	5.3.11.1.2
| | | |-partner	4	0..1	switch	5.3.11.1.3
|-controller	1	0+	OpenDRIVE	5.4
| |-control	2	1+	controller	5.4.1
|-junction	1	0+	OpenDRIVE	5.5
| |-connection	2	1+	junction	5.5.1
| | |-laneLink	3	0+	connection	5.5.1.1
| |-priority	2	0+	junction	5.5.2
| |-controller	2	0+	junction	5.5.3
|-junctionGroup	1	0+	OpenDRIVE	5.6
| |-junctionReference	2	1+	junctionGroup	5.6.1
|-station	1	0+	OpenDRIVE	5.7
| |-platform	2	1+	station	5.7.1
| | |-segment	3	1+	platform	5.7.1.1

At any level, there may be the following beads:

bead name	level	instances per parent	parent	chapter
userData	any	0+	any	5.8
include	any	0+	any	5.9
set	any	0+	any	5.10
|-instance	set+1	1+	set	5.10.1

5 File Entries

5.1 Enclosing Tag

The overall enclosing tag of the file is:

delimiters	`<OpenDRIVE>...</OpenDRIVE>`
instances	`1`
attributes	`none`

5.2 Header

The header record is the very first record within the `OpenDRIVE` tag.

delimiters	`<header>...</header>`
parent	`<OpenDRIVE>`
instances	`1`
attributes	

name	***type***	***unit***	***value***	***description***
`revMajor`	ushort	-	1	major revision number of OpenDRIVE® format
`revMinor`	ushort	-	4	minor revision number of OpenDRIVE® format
`name`	string	-	-	database name
`version`	float	-	-	version number of this database (format: a.bb)
`date`	string	-	-	time/date of database creation
`north`	double	m	-	maximum inertial y value
`south`	double	m	-	minimum inertial y value
`east`	double	m	-	maximum inertial x value
`west`	double	m	-	minimum inertial x value
`vendor`	string	-	-	vendor name

5.2.1 Geo Reference

The information for geographic reference of a database may be provided as child node to the OpenDRIVE *header* node. It will provide all information necessary to convert OpenDRIVE's cartesian x/y/z co-ordinates into a corresponding geographic reference system. There must be no more than one definition of the geographic projection.

If the definition is missing, the cartesian co-ordinates will be the only ones useable within the database.

delimiters	`<geoReference>...</geoReference>`
parent	`<header>`
instances	`0..1`
attributes	*none*

The projection string according to OGC WKT (see also http://spatialreference.org) shall be given as *CDATA* child node of the `<geoReference>` node since it may contain characters which interfere with the XML syntax of a node's attribute.

Example (see also http://spatialreference.org/ref/epsg/32632/ogcwkt/):

```
<geoReference>
  <![CDATA[PROJCS["WGS 84 / UTM zone 32N",GEOGCS["WGS
  84",DATUM["WGS_1984",SPHEROID["WGS
  84",6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG
  ","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["deg
  ree",0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG"
  ,"4326"]],UNIT["metre",1,AUTHORITY["EPSG","9001"]],PROJECTION["Tran
  sverse_Mercator"],PARAMETER["latitude_of_origin",0],PARAMETER["cent
  ral_meridian",9],PARAMETER["scale_factor",0.9996],PARAMETER["false_
  easting",500000],PARAMETER["false_northing",0],AUTHORITY["EPSG","32
  632"],AXIS["Easting",EAST],AXIS["Northing",NORTH]]]]>
</geoReference>
```

5.3 Road Records

Roads are the principal containers of information within a database. For an overview of records which can be stored within a road, see chapter 4.7.

5.3.1 Road Header Record

The road header record defines the basic parameters of an individual road. It is followed immediately by other records defining geometry and logical properties of the road.

delimiters	`<road>...</road>`
parent	`<OpenDRIVE>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	name of the road
`length`	double	m	-	total length of the reference line in the xy-plane
`id`	string	-	- [0,∞[	unique ID within database (preferably an integer number, uint32_t)
`junction`	string	m	- `-1`	ID of the junction to which the road belongs as a connecting road (= -1 for none)

5.3.2 Road Link Record

This record follows the Road Header Record if the road is (as usual) linked to a successor, a predecessor, or a neighbor (see children of the link record). Isolated roads may omit this record.

delimiters	`<link>...</link>`
parent	`<road>`
instances	`0..1`
attributes	`none`

5.3.2.1 Road Predecessor

This record provides detailed information about the predecessor of a road. The predecessor may be of type road or junction.

delimiters	`<predecessor.../>`
parent	`<link>`
instances	`0..1`
attributes	

name	***type***	***unit***	***value***	***description***
`elementType`	string	-	`road` `junction`	type of the linked element
`elementId`	string	-	-	ID of the linked element
`contactPoint`	string	-	`start` `end`	contact point of link on the linked element

5.3.2.2 Road Successor

This record provides detailed information about the successor of a road. The successor may be of type road or junction.

delimiters	`<successor.../>`
parent	`<link>`
instances	`0..1`
attributes	

name	***type***	***unit***	***value***	***description***
`elementType`	string	-	`road` `junction`	type of the linked element
`elementId`	string	-	-	ID of the linked element
`contactPoint`	string	-	`start` `end`	contact point of link on the linked element

5.3.2.3 Road Neighbor

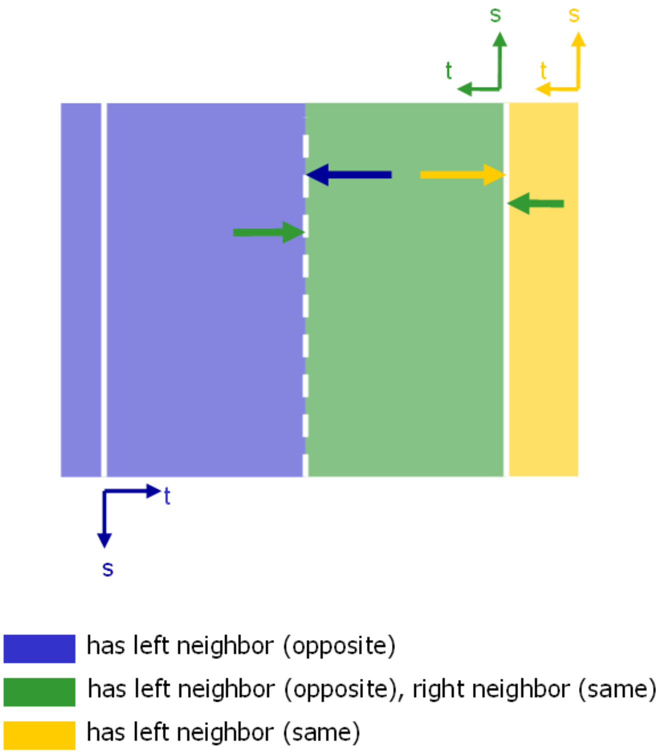
This record provides detailed information about the neighbor of a road. The neighbor must be of type road.

Note: The neighbor entry has been introduced mainly for legacy purposes. For the design of new road networks, it is recommended to define both driving directions of a road along a single reference line and to avoid using the neighbor entry (this recommendation applies outside junctions only; within junctions, single reference lines may and should be used for each driving direction, however).

delimiters	`<neighbor.../>`
parent	`<link>`
instances	`0..2`
attributes	

name	***type***	***unit***	***value***	***description***
`side`	string	-	`left` `right`	information which neighbor is being configured
`elementId`	string	-	-	ID of the linked road
`direction`	string	-	`same` `opposite`	direction of the neighbor relative to own direction

Illustration:


s
s
t
t
t
s
🟦 has left neighbor (opposite)
🟩 has left neighbor (opposite), right neighbor (same)
🟨 has left neighbor (same)

5.3.3 Road Type Record

The type of a road may change over its entire length. Therefore, a separate record is provided for the definition of the road type with respect to a certain section of the road. A road type entry is valid for the entire cross section of the road. It is also valid until a new road type record is provided or the road ends.

delimiters	`<type.../>`
parent	`<road>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	start position (s-coordinate)
`type`	string	-	see 6.1	for supported types, see chapter 6.1

5.3.3.1 Speed Record

This record defines the default maximum speed allowed in conjunction with the specified road type.

delimiters	`<speed.../>`
parent	`<type>`
instances	`0..1`
attributes	

name	***type***	***unit***	***value***	***description***

`max`	string	m/s	*`no limit`*
			`undefined`
			[0,∞[
				maximum allowed speed, given as string or numerical value in the respective unit (see next attribute)

`unit`	string	-	see 2.2	unit of the attribute `max` (optional,)

5.3.4 *Road Plan View Record*

The plan view record contains a series of geometry records which define the layout of the road's reference line in the x/y-plane (plan view).

delimiters	`<planView>...</planView>`
parent	`<road>`
instances	`1`
attributes	`none`

5.3.4.1 Road Geometry Header Record

A sequence of road geometry records defines the layout of the road's reference line in the in the x/y-plane (plan view). The geometry records must occur in ascending order (i.e. increasing s-position). The geometry information is split into a header which is common to all geometric elements and a sub-sequent bead containing the actual geometric element’s data (depending on the type of geometric element).

Currently, four types of geometric elements are supported:

- straight lines
- spirals
- arcs
- cubic polynomials

delimiters	`<geometry>...</geometry>`
parent	`<planView>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	start position (s-coordinate)
`x`	double	m	]-∞,∞[	start position (x inertial)
`y`	double	m	]-∞,∞[	start position (y inertial)
`hdg`	double	rad	]-∞,∞[	start orientation (inertial heading)
`length`	double	m	[0,∞[	length of the element's reference line

This record is followed immediately by a record with more information about the actual geometry element.

5.3.4.1.1 Geometry, Line Record

This record describes a straight line as part of the road's reference line.

delimiters	`<line.../>`
parent	`<geometry>`
instances	`1`
attributes	`none`

5.3.4.1.2 Geometry, Spiral Record (Clothoids)

This record describes a spiral as part of the road's reference line. For this type of spiral, the curvature change between start and end of the element is linear.

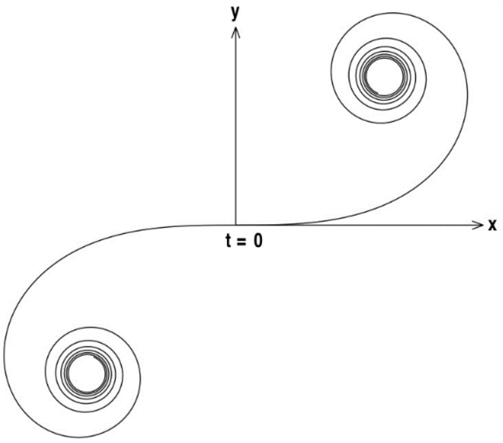
In order to provide consistency between spiral evaluations of various implementations, users are highly recommended to use identical algorithms, or, at least, algorithms with reasonable accuracy within the applicable range (i.e. curvature, length of spiral etc.).

The theory of clothoids is described (for example) at

http://en.wikipedia.org/wiki/Euler_spiral

A library for computing clothoids can be downloaded at

http://www.opendrive.org/downloads/spiral.zip



delimiters	`<spiral.../>`
parent	`<geometry>`
instances	`1`
attributes	

name	***type***	***unit***	***value***	***description***
`curvStart`	double	1/m	]-∞,∞[	curvature at the start of the element
`curvEnd`	double	1/m	]-∞,∞[	curvature at the end of the element

5.3.4.1.3 Geometry, Arc Record

This record describes an arc as part of the road’s reference line.

delimiters `<arc.../>`
parent `<geometry>`
instances `1`
attributes

name	***type***	***unit***	***value***	***description***
`curvature`	double	1/m	]-∞,∞[	constant curvature throughout the element

5.3.4.1.4 Geometry, Cubic Polynomial Record

This record describes a cubic polynomial as part of the road's reference line. The polynomial is described in the local u/v co-ordinate system of the starting point (with u pointing in the local s direction and v pointing in the local t direction). Each local co-ordinate is calculated by:

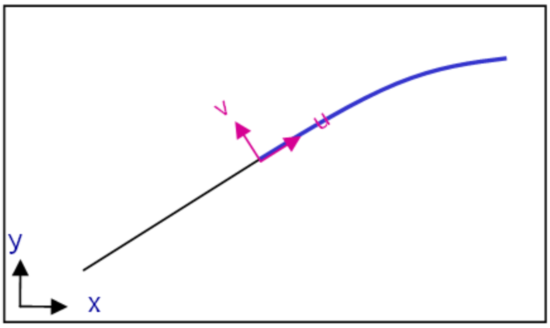
$$v_{local} = a + b*du + c*du^2 + d*du^3$$

The conversion of u/v co-ordinates into x/y co-ordinates can be performed easily by simple geometric transformations (i.e. one translation and one rotation) relative to the starting point

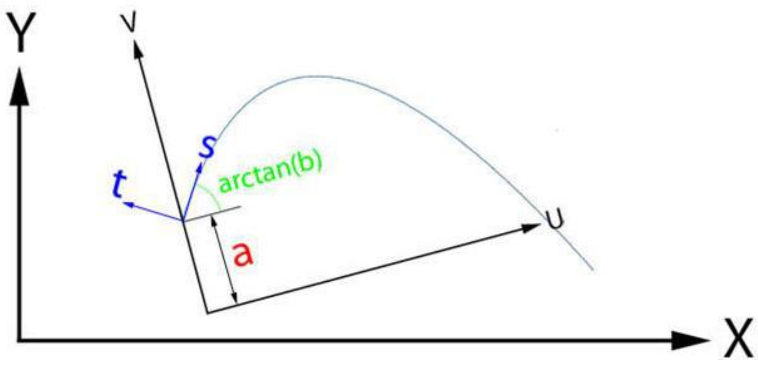
delimiters	`<poly3.../>`
parent	`<geometry>`
instances	1
attributes	

name	type	unit	value	description
a	double	m	]-∞,∞[	parameter A
b	double	1	]-∞,∞[	parameter B
c	double	1/m	]-∞,∞[	parameter C
d	double	1/m²	]-∞,∞[	parameter D

Illustration:



NOTE: In order to stick to the start point and orientation according to the <geometry> node, the parameters a and b will have to be zero. Providing non-zero values for these parameters will lead to a shift and rotation of the s/t co-ordinates according to the following figure (image courtesy of VTI):



5.3.4.1.5 Geometry, Parametric Cubic Curve Record

This record describes a parametric cubic curve as part of the road's reference line **in a local u/v co-ordinate system**. The curve is composed of two polynomials of third order, describing local u- and v-coordinates in reference to a common parameter (p). Unless otherwise stated, the parameter p is in the range [0;1]. Alternatively, it may be given in the range [0;arc length]. Any deviation from the default rule has to be stated with the attribute "pRange" and the parameters of the polynomials have to be adapted accordingly.

Each co-ordinate is calculated by:

$$u_{local} = a_u + b_u*p + c_u*p^2 + d_u*p^3$$
$$v_{local} = a_v + b_v*p + c_v*p^2 + d_v*p^3$$

The conversion from the local u/v co-ordinates into the inertial x/y co-ordinates is performed as described in the previous chapter (cubic polynomial record).

delimiters	`<paramPoly3.../>`
parent	`<geometry>`
instances	1
attributes	

name	***type***	***unit***	***value***	***description***
aU	double	m	]-∞,∞[	parameter A for u
bU	double	1	]-∞,∞[	parameter B for u
cU	double	1/m	]-∞,∞[	parameter C for u
dU	double	1/m²	]-∞,∞[	parameter D for u
aV	double	m	]-∞,∞[	parameter A for v
bV	double	1	]-∞,∞[	parameter B for v
cV	double	1/m	]-∞,∞[	parameter C for v
dV	double	1/m²	]-∞,∞[	parameter D for v
pRange	string	-	arcLength normalized	range / scope of parameter p

5.3.5 Road Elevation Profile Record

The elevation profile record contains a series of elevation records which define the characteristics of the road's elevation along the reference line.

delimiters `<elevationProfile>...</elevationProfile>`
parent `<road>`
instances `0..1`
attributes `none`

5.3.5.1 Road Elevation Record

The elevation record defines an elevation entry at a given reference line position. Entries must be defined in increasing order along the reference line. The parameters of an entry are valid until the next entry starts or the road's reference line ends. Per default, the elevation of a road is zero.

The elevation is stored in a polynomial function of third order. It looks like:

$$elev = a + b*ds + c*ds^2 + d*ds^3$$

with

`elev` being the elevation (inertial z) at a given position
`a, b, c, d` being the coefficients and
`ds` being the distance along the reference line between the start of the entry and the actual position.

Therefore, ds starts at zero for each entry. The absolute position of an elevation value is calculated by

$$s = s_{start} + ds$$

with

s being the absolute position (track co-ordinate system)
s_{start} being the start position of the entry in the track co-ordinate system
ds being the delta between the start position and the requested position

delimiters `<elevation.../>`
parent `<elevationProfile>`
instances `1+`
attributes

name	type	unit	value	description
s	double	m	[0,∞[	start position (s-coordinate)
a	double	m	]-∞,∞[	parameter A, elevation at s=0
b	double	1	]-∞,∞[	parameter B
c	double	1/m	]-∞,∞[	parameter C
d	double	$1/m^2$	]-∞,∞[	parameter D

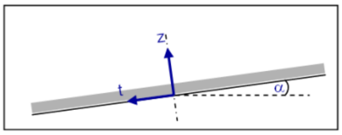
5.3.6 Road Lateral Profile Record

The lateral profile record contains a series of superelevation and crossfall records which define the characteristics of the road surface's banking along the reference line.

delimiters	`<lateralProfile>...</lateralProfile>`
parent	`<road>`
instances	`0..1`
attributes	`none`

5.3.6.1 Road Superelevation Record

The superelevation of the road is defined as the road section’s roll angle around the s-axis (superelevation is positive for road surfaces “falling” to the right side, i.e. the following figure shows a negative superelevation).



Each superelevation record defines an entry at a given reference line position. Entries must be defined in increasing order along the reference line. The parameters of an entry are valid until the next entry starts or the road’s reference line ends. Per default, the superelevation of a road is zero.

The superelevation is stored in a polynomial function of third order. It looks like:

$$sElev = a + b*ds + c*ds^2 + d*ds^3$$

with

`sElev`	being the superelevation at a given position, default: `sElev = 0`
`a, b, c, d`	being the coefficients and
`ds`	being the distance along the reference line between the start of the entry and the actual position.

Therefore, ds starts at zero for each entry. The absolute position of a superelevation value is calculated by

$$s = s_{start} + ds$$

with

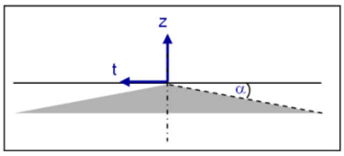
s	being the absolute position (track co-ordinate system)
s_{start}	being the start position of the entry in the track co-ordinate system
ds	being the delta between the start position and the requested position

delimiters	`<superelevation.../>`
parent	`<lateralProfile>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	start position (s-coordinate)
`a`	double	rad	]-∞,∞[	parameter A, superelevation at s=0
`b`	double	rad/m	]-∞,∞[	parameter B
`c`	double	rad/m²	]-∞,∞[	parameter C
`d`	double	rad/m³	]-∞,∞[	parameter D

5.3.6.2 Crossfall Record

The crossfall of the road is defined as the road surface´s angle relative to the t-axis. Positive crossfall results in a road surface “falling” from the reference line to the outer boundary.



The crossfall can be defined per side of the road. It will be applied to each lane of the respective side which is not being explicitly excluded from its application.

The crossfall is stored in a polynomial function of third order. It looks like:

$$crfall = a + b*ds + c*ds^2 + d*ds^3$$

with

crfall being the crossfall at a given position, default: crfall = 0
a, b, c, d being the coefficients and
ds being the distance along the reference line between the start of the entry and the actual position.

Therefore, ds starts at zero for each entry. The absolute position of a crossfall value is calculated by

$$s = s_{start} + ds$$

with

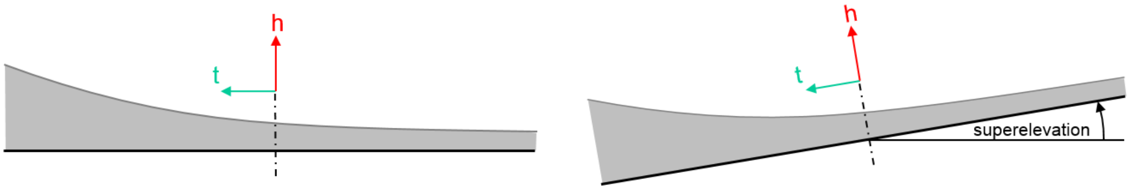
s being the absolute position (track co-ordinate system)
s_{start} being the start position of the entry in the track co-ordinate system
ds being the delta between the start position and the requested position

delimiters <crossfall.../>
parent <lateralProfile>
instances 0+
attributes

name	type	type	value	description
side	string	string	left right both	applicable side of the road
s	double	m	[0,∞[	start position (s-coordinate)
a	double	rad	]-∞,∞[	parameter A, crossfall at s=0
b	double	rad/m	]-∞,∞[	parameter B
c	double	rad/m²	]-∞,∞[	parameter C
d	double	rad/m³	]-∞,∞[	parameter D

5.3.6.3 Road Shape Record

The shape of the road is defined as the road section's surface relative to the reference plane (which may itself be tilted around the s-axis due to an existing super-elevation). This shape may be described as a series of 3rd order polynomials for a given "s" station. Multiple shapes may be defined along a road at consecutive "s" stations. Between the shapes at these stations, there shall be a linear interpolation along s.



The shape polynomial looks like:

$$h_{Shape} = a + b*dt + c*dt^2 + d*dt^3$$

with

h_{Shape} being the height above the reference plane at a given position, default: $h_{Shape} = 0$

a, b, c, d being the coefficients and

dt being the distance perpendicular to the reference line between the start of the entry and the actual position.

Therefore, dt starts at zero for each entry. The absolute position of a shape value is calculated by

$$t = t_{start} + dt$$

with

t being the absolute position (track co-ordinate system)

t_{start} being the start position of the entry in the track co-ordinate system

dt being the delta between the start position and the requested position

delimiters	`<shape.../>`
parent	`<lateralProfile>`
instances	0+

attributes

name	***type***	***unit***	***value***	***description***
s	double	m	[0,∞[	start position (s-coordinate)
t	double	m	]-∞,∞[	start position (t-coordinate)
a	double	m	]-∞,∞[	parameter A, relative height at t=0
b	double	1	]-∞,∞[	parameter B
c	double	1/m	]-∞,∞[	parameter C
d	double	1/m²	]-∞,∞[	parameter D

5.3.7 *Road Lanes Record*

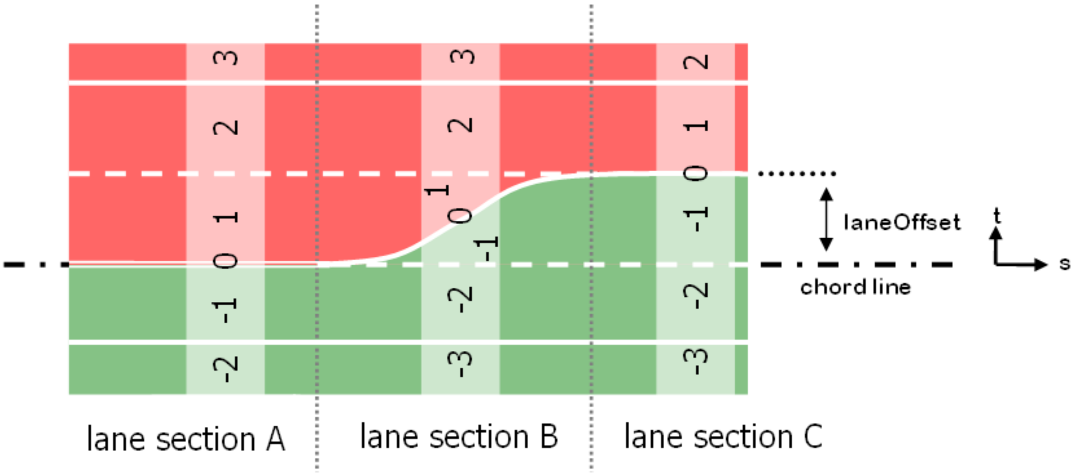
The lanes record contains a series of lane section records which define the characteristics of the road cross sections with respect to the lanes along the reference line.

delimiters	`<lanes>...</lanes>`
parent	`<road>`
instances	`1`
attributes	`none`

5.3.7.1 Road Lane Offset Record

The lane offset record defines a lateral shift of the lane reference line (which is usually identical to the road reference line). This may be used for an easy implementation of a (local) lateral shift of the lanes relative to the road’s reference line. Especially the modeling of inner-city layouts or "2+1" cross-country road layouts can be facilitated considerably by this feature.

The following figure illustrates the modeling of additional "inner" lanes along a straight reference line:



The actual offset at a given point is computed with a polynomial function of third order. It looks like:

$$\texttt{offset = a + b*ds + c*ds}^2\texttt{ + d*ds}^3$$

with

`offset`	being the lateral offset at a given position
`a, b, c, d`	being the coefficients and
`ds`	being the distance along the reference line between the start of the entry and the actual position.

Therefore, ds starts at zero for each entry. The absolute position of a width value is calculated by

$$s = s_{start} + ds$$

with

s	being the absolute position (track co-ordinate system)
s_{start}	being the start position of the lateral offset entry in the track co-ordinate system

`ds` being the delta between the start position (s_{start}) and the requested position

A new lane offset entry is required each time the polynomial function changes.

delimiters `<laneOffset.../>`
parent `<lanes>`
instances `0+`
attributes

name	**type**	**unit**	**value**	**description**
s	double	m	[0,∞[	start position (s-coordinate)
a	double	m	]-∞,∞[	parameter A, offset at s=0
b	double	1	]-∞,∞[	parameter B
c	double	1/m	]-∞,∞[	parameter C
d	double	1/m²	]-∞,∞[	parameter D

5.3.7.2 Road Lane Section Record

The lane section record defines the characteristics of a road cross-section. It specifies the numbers and types of lanes and further features of the lanes. At least one record must be defined in order to use a road. A lane section record is valid until a new lane section record is defined. If multiple lane section records are defined, they must be listed in ascending order.

The actual lanes and their properties are children of the lane section record and the lane records, respectively.

delimiters `<laneSection>...</laneSection>`
parent `<lanes>`
instances `1+`
attributes

name	**type**	**unit**	**value**	**description**
s	double	m	-	start position (s-coordinate)
singleSide	string	-	true false	lane section entry is valid for one side only (left or right, depending on the child entries)

For the naming convention of lanes, see chapter 3.2.

5.3.7.2.1 Left / Center / Right Records

For easier navigation through a road description, the lanes under a lane section are grouped into left, center and right lanes. At least one entry (left, center or right) must be present.

delimiters	`<left>...</left>`
	`<center>...</center>`
	`<right>...</right>`

parent	`<laneSection>`
instances	`1`
attributes	`none`

5.3.7.2.1.1 Lane Record

Lane records are found within the left/center/right records. They define the IDs of the actual lanes (and, therefore, their position on the road, see conventions in 3). In order to prevent confusion, lane records should represent the lanes from left to right (i.e. with descending ID). All properties of the lanes are defined as children of the lane records.

delimiters	`<lane>...</lane>`
parent	`<left> / <center> / <right>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`id`	int	-	]-∞,∞[	id of the lane (according to convention)
`type`	string	-	see 6.5	type of the lane
`level`	string	string	`true` `false`	"`true`" = keep lane on level, .i.e. do not apply superelevation or crossfall "`false`" = apply superelevation and crossfall to this lane (default, also used if argument `level` is missing) lanes are also kept on level if the argument `level` is present but no superelevation or crossfall have been defined.

5.3.7.2.1.1.1 Lane Link Record

In order to facilitate navigation through a road network on a per-lane basis, lanes should be provided with predecessor/successor information. Only when lanes end at a junction or have no physical link, this record should be omitted.

For links between lanes on the same physical road (i.e. identical reference line), the lane predecessor/ successor information provides the ids of lanes on the preceding or following lane section. For links between lanes on different roads, i.e. roads directly connecting to each other without a junction, the complete link information must be composed from the corresponding road link record and the lane link record.

delimiters	`<link>...</link>`
parent	`<lane>`
instances	`0..1`
attributes	`none`

5.3.7.2.1.1.1.1 Lane Predecessor

This record provides detailed information about the predecessor of a lane.

delimiters	`<predecessor.../>`
parent	`<link>`
instances	`0..1`
attributes	

name	***type***	***unit***	***value***	***description***
`id`	int	-	]-∞,∞[	ID of the linked lane

5.3.7.2.1.1.1.2 Lane Successor

This record provides detailed information about the successor of a lane.

delimiters	`<successor.../>`
parent	`<link>`
instances	`0..1`
attributes	

name	***type***	***unit***	***value***	***description***
`id`	int	-	]-∞,∞[	ID of the linked lane

5.3.7.2.1.1.2 Lane Width Record

Each lane within a road's cross section can be provided with several width entries. At least one entry must be defined for each lane, except for the center lane which is, per convention, of zero width (see 3.2). Each entry is valid until a new entry is defined. If multiple entries are defined for a lane, they must be listed in ascending order.

The actual width at a given point is computed with a polynomial function of third order. It looks like:

$$\mathtt{width = a + b*ds + c*ds^2 + d*ds^3}$$

with

`width` being the width at a given position
`a, b, c, d` being the coefficients and
`ds` being the distance along the reference line between the start of the entry and the actual position.

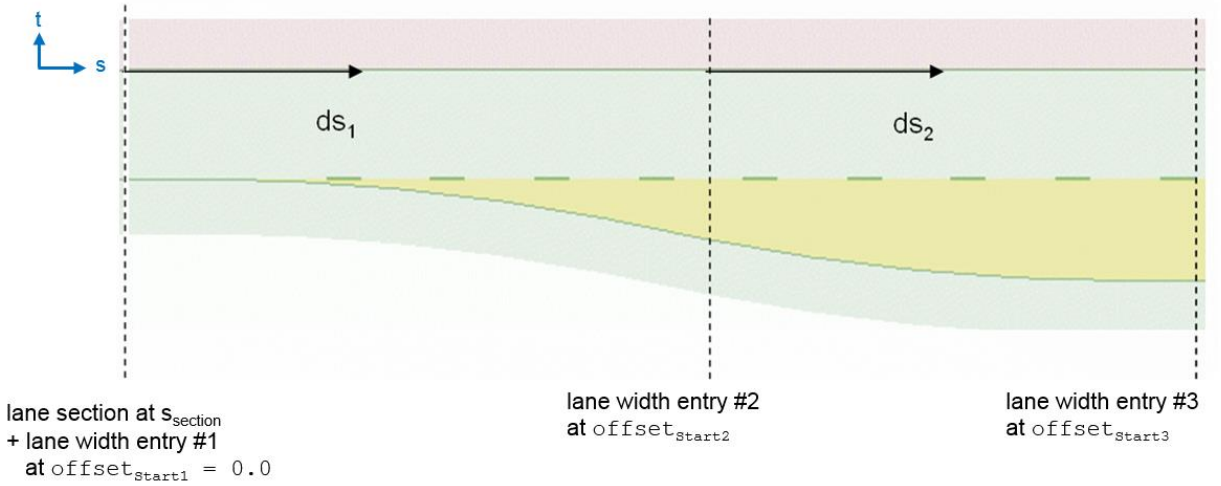
Therefore, ds starts at zero for each entry. The absolute position of a width value is calculated by

$$\mathtt{s = s_{section} + offset_{start} + ds}$$

with

$\mathtt{s}$ being the absolute position (track co-ordinate system)
$\mathtt{s_{section}}$ being the start position of the preceding lane section record (see 0) in the track co-ordinate system
$\mathrm{offset}_{\mathtt{start}}$ being the offset of the entry relative to the preceding lane section record
`ds` being the delta between the offset ($\mathtt{s_{start}}$) and the requested position

The following figure illustrates this convention for a lane with varying width over a given range:


t
s
ds_1
ds_2
lane section at $s_{section}$
+ lane width entry #1
at $\mathtt{offset_{start1} = 0.0}$
lane width entry #2
at $\mathtt{offset_{start2}}$
lane width entry #3
at $\mathtt{offset_{start3}}$

A new width entry is required each time the polynomial function changes.

delimiters `<width.../>`
parent `<lane>`
instances `1+ (if not <border> entry is present)`
restrictions `not allowed for center lane (laneId=0)`
attributes

name	***type***	***unit***	***value***	***description***
`sOffset`	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *`laneSection`* record
`a`	double	m	]-∞,∞[	parameter A, width at s=0
`b`	double	1	]-∞,∞[	parameter B
`c`	double	1/m	]-∞,∞[	parameter C
`d`	double	1/m²	]-∞,∞[	parameter D

5.3.7.2.1.1.3 Lane Border Record

Instead of describing lanes by their width entries and, thus, invariably depending on influences of inner lanes on outer lanes, it might be more convenient to just describe the outer border of each lane independent of any inner lanes' parameters. Especially in cases where road data is derived from measurements, this type of definition will provide a more convenient method without the need to tesselate road sections into too many parts.

Width and border records are, of course, mutually exclusive for identical areas of a lane. If for a lane both entries (width and border) are defined, only the width entry shall prevail. If you are using border definitions, then also the definition of an offset record (see "Road Lane Offset Record") becomes obsolete.

~~At least one entry must be defined for each lane, except for the center lane which is, per convention, of zero width (see 3.2). Each entry is valid until a new entry is defined. If multiple entries are defined for a lane, they must be listed in ascending order.~~

The actual width at a given point is computed with a polynomial function of third order. It looks like:

$$t_{border} = a + b*ds + c*ds^2 + d*ds^3$$

with

t_{border} being the t-position of the border at a given ds position
a, b, c, d being the coefficients and
ds being the distance along the reference line between the start of the entry and the actual position.

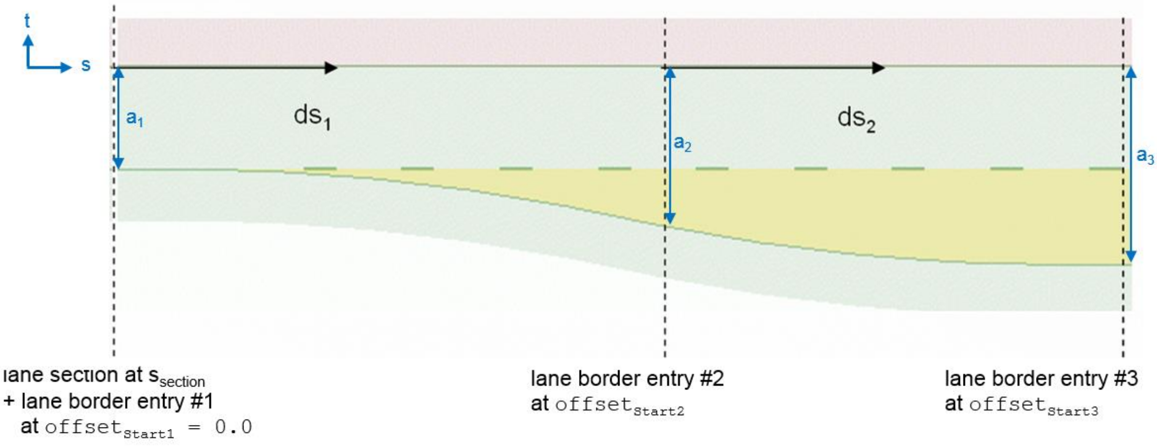
Therefore, ds starts at zero for each entry. The absolute position of a width value is calculated by

$$s = s_{section} + offset_{start} + ds$$

with

s being the absolute position (track co-ordinate system)
$s_{section}$ being the start position of the preceding lane section record (see 0) in the track co-ordinate system
$offset_{start}$ being the offset of the entry relative to the preceding lane section record
ds being the delta between the offset (s_{start}) and the requested position

The following figure illustrates this convention for a lane with varying border shape over a given range:



A new border entry is required each time the polynomial function changes.

delimiters	`<border.../>`
parent	`<lane>`
instances	`1+ (if not <width> entry is present)`
attributes	

name	type	unit	value	description
sOffset	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *laneSection* record
a	double	m	]-∞,∞[	parameter A, border position at s=0
b	double	1	]-∞,∞[	parameter B
c	double	1/m	]-∞,∞[	parameter C
d	double	1/m²	]-∞,∞[	parameter D

5.3.7.2.1.1.4 Road Mark Record

Each lane within a road cross section can be provided with several road mark entries. The road mark information defines the style of the line at the lane's outer border. For left lanes, this is the left border, for right lanes the right one. The style of the line separating left and right lanes is determined by the road mark entry for lane zero (i.e. the center lane)

delimiters	`<roadMark>...</roadMark>`
parent	`<lane>`
instances	`0+`
attributes	

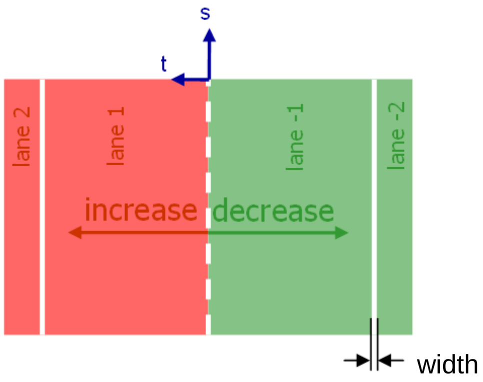
name	***type***	***unit***	***value***	***description***
`sOffset`	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *laneSection* record
`type`	string	-	see 6.2	type of the road mark
`weight`	string	-	see 6.3	weight of the road mark
`color`	string	-	see 6.4	color of the road mark
`material`	string	-	string	material of the road mark (identifiers to be defined, use "standard" for the moment.
`width`	double	m	[0,∞[	width of the road mark – optional

`laneChange`	string	-	`increase`
			`decrease`
			`both`
			`none`
				allow a lane change in the indicated direction taking into account that lanes are numbered in ascending order from right to left. If the attribute is missing, "both" is assumed to be valid.

The parameter `weight` may be used for a categorized definition of the width of a road mark (e.g. "standard" and "bold" according to the corresponding country's design rules) whereas the optional parameter `width` may be used for an exact definition of an individual road mark's width. This may be required e.g. in cases which deviate from a common design rule.

For an exact evaluation of a road mark's borders including the width of the mark, the convention shall apply that a road mark's centerline is always positioned on the respective lane's outer border line (so that the outer half of the road mark is physically placed on the next lane).

Illustration:



5.3.7.2.1.1.4.1 Road Mark Type

Road marks may be further described in terms of their type. Instead of providing additional parameters to the `roadMark` tag, type definitions shall be defined as children to the `roadMark` tag. Each `type` definition must contain one or more `line` definitions with additional information about the lines the road mark is composed of.

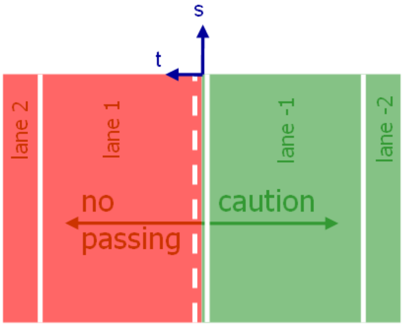
delimiters `<type>...</type>`
parent `<roadMark>`
instances `0..1`
attributes

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	name of the road mark type
`width`	double	m	[0,∞[	line width

5.3.7.2.1.1.4.1.1 Road Mark Type – Line Definition

A road mark may consist of one or more elements. Multiple elements will usually be positioned side by side.

Currently, the elements can only be lines. Each line can be defined by its line/space setup and by a lateral offset to the road mark's reference position (i.e. the border between two lanes). A line definition is valid for a given length (i.e. the total of visible line and invisible space) and will be repeated automatically.



delimiters `<line.../>`
parent `<type>`
instances `1+`
attributes

name	***type***	***unit***	***value***	***description***
`length`	double	m	[0,∞[	length of the visible part
`space`	double	m	[0,∞[	length of the space following the visible part
`tOffset`	double	m	]-∞,∞[	lateral offset from the lane border
`sOffset`	double	m	[0,∞[	initial longitudinal offset of the line definition from the start of the road mark definition

`rule`	string	-	`no passing`
			`caution`
			`none`
				rule which is to be observed when passing the line from inside

`width`	double	m	[0,∞[	line width; if given, supersedes the definition in the parent tag

5.3.7.2.1.1.5 Lane Material Record

Each lane within a road cross section may be provided with several entries defining its material. Each entry is valid until a new entry is defined. If multiple entries are defined, they must be listed in increasing order.

delimiters	`<material.../>`
parent	`<lane>`
instances	`0+`
restrictions	`not allowed for center lane (laneId=0)`
attributes	

name	***type***	***unit***	***value***	***description***
`sOffset`	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *`laneSection`* record
`surface`	string	-	-	surface material code, depending on application
`friction`	double	-	[0,∞[	friction value
`roughness`	double	-	[0,∞[	roughness (e.g. for sound and motion systems)

5.3.7.2.1.1.6 Lane Visibility Record

Each lane within a road cross section may be provided with several entries defining the visibility in four directions relative to the lane's direction. Each entry is valid until a new entry is defined. If multiple entries are defined, they must be listed in increasing order.

For left lanes (positive ID), the forward direction is oriented opposite to the track's direction, for right lanes, the forward direction and the track's direction are identical.

delimiters	`<visibility.../>`
parent	`<lane>`
instances	`0+`
restrictions	`not allowed for center lane (laneId=0)`
attributes	

name	***type***	***unit***	***value***	***description***
`sOffset`	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *`laneSection`* record
`forward`	double	m	[0,∞[	visibility in forward direction
`back`	double	m	[0,∞[	visibility in reverse direction
`left`	double	m	[0,∞[	visibility to the left
`right`	double	m	[0,∞[	visibility to the right

5.3.7.2.1.1.7 Lane Speed Record

This record defines the maximum allowed speed on a given lane. Each entry is valid in direction of the increasing s co-ordinate until a new entry is defined. If multiple entries are defined, they must be listed in increasing order.

delimiters	`<speed.../>`
parent	`<lane>`
instances	`0+`
restrictions	`not allowed for center lane (laneId=0)`
attributes	

name	***type***	***unit***	***value***	***description***
`sOffset`	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *`laneSection`* record
`max`	double	m/s	[0,∞[	maximum allowed speed
`unit`	string	-	see 2.2	unit of the attribute max (optional)

5.3.7.2.1.1.8 Lane Access Record

This record defines access restrictions for certain types of road users. The record can be used to complement restrictions resulting from signs or signals in order to control the traffic flow in a scenario. Each entry is valid in direction of the increasing s co-ordinate until a new entry is defined. If multiple entries are defined, they must be listed in increasing order.

delimiters	`<access.../>`
parent	`<lane>`
instances	`0+`
restrictions	`not allowed for center lane (laneId=0)`
attributes	

name	***type***	***unit***	***value***	***description***
`sOffset`	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *`laneSection`* record
`restriction`	string	-	see 6.9	identifier of the participant to which the restriction applies

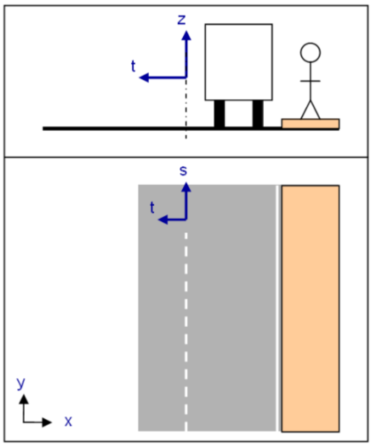
5.3.7.2.1.1.9 Lane Height Record

The surface of a lane may be offset from the plane defined by the reference line and the corresponding elevation and crossfall entries (e.g. pedestrian walkways are typically a few centimeters above road level). The height record provides a simplified method to describe this offset by setting an inner and outer offset from road level at discrete positions along the lane profile.

delimiters	`<height.../>`
parent	`<lane>`
instances	`0+`
restrictions	`not allowed for center lane (laneId=0)`
attributes	

name	***type***	***unit***	***type***	***description***
`sOffset`	double	m	[0,∞[	start position (s-coordinate) relative to the position of the preceding *`laneSection`* record
`inner`	double	m	]-∞,∞[	inner offset from road level
`outer`	double	m	]-∞,∞[	outer offset from road level

Illustration:



5.3.8 *Road Objects Record*

The objects record is the container for all objects along a road.

delimiters	`<objects>...</objects>`
parent	`<road>`
instances	`0..1`
attributes	`none`

5.3.8.1 Object Record

The object record has been introduced for application-dependent elements. It is very flexible due to several parameters that can be used to provide the application with additional information. The most frequently used types of objects may become part of the OpenDRIVE® specification in future releases.

delimiters	`<object>...</object>`
parent	`<objects>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`type`	string	-	-	type of the object (for a parking space, use the identifier `"parkingSpace"`)
`name`	string	-	-	name of the object
`id`	string	n/a	-	unique ID within database
`s`	double	m	[0,∞[	track position of object's origin (s-position)
`t`	double	m	]-∞,∞[	track position of object's origin (t-position)
`zOffset`	double	m	]-∞,∞[	z offset of object's origin from track level
`validLength`	double	m	[0,∞[	extent of object's validity along s-axis (0.0 for point object)

`orientation`	string	-	`+`
			`-`
			`none`
				"+" = valid in positive track direction
				"-" = valid in negative track direction
				"none" = valid in both directions

`length`	double	m	[0,∞[	length of the object (local dx)
`width`	double	m	[0,∞[	width of the object (local dy)
`radius`	double	m	[0,∞[	radius of the object; alternative to width and length
`height`	double	m	]-∞,∞[	height of the object (local dz)
`hdg`	double	rad	]-∞,∞[	heading angle of the object relative to road direction
`pitch`	double	rad	]-∞,∞[	pitch angle of the object relative to road pitch
`roll`	double	rad	]-∞,∞[	roll angle of the object relative to road roll

Illustration for circular (or cylindric) objects:

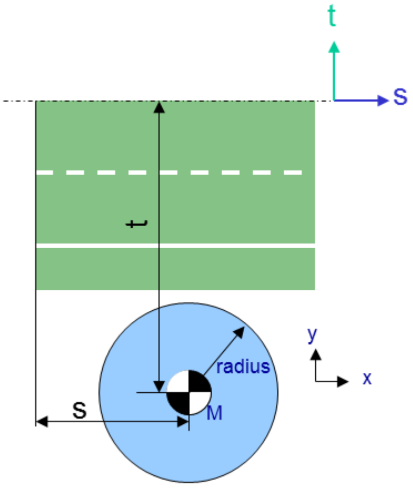
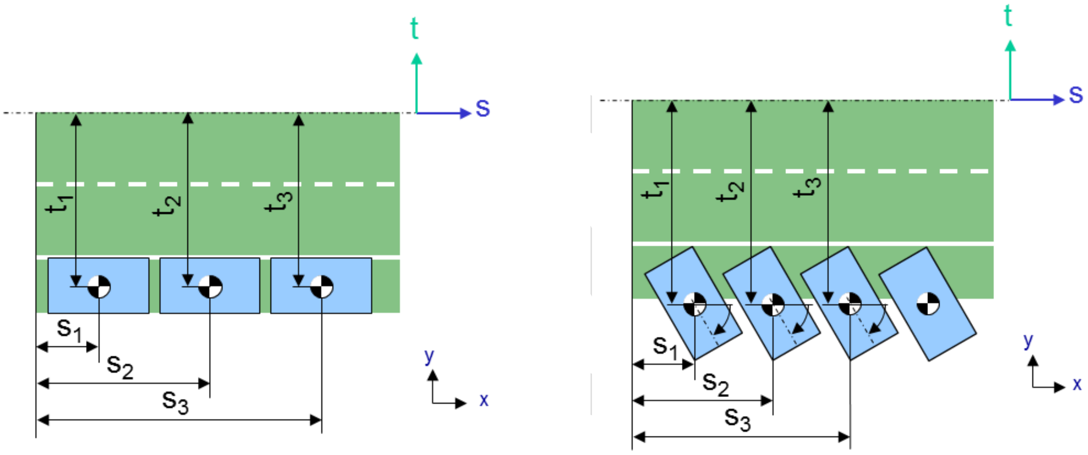


Illustration for parking spaces (aligned with road and with offset angle):



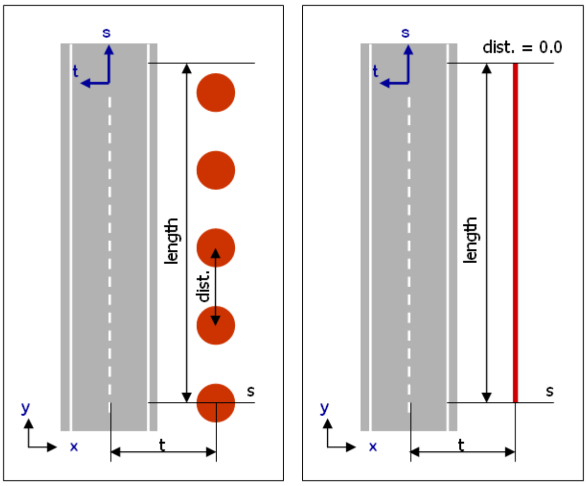
5.3.8.1.1 Object Repeat Record

In order to avoid multiple definitions for multiple instances of the same object, repeat parameters may be defined for the original object. Some of the parameters in the object node may be overwritten in order to specify linear transitions of e.g. the lateral offset of an object. For attributes which are not specified in the object repeat record, the attributes of the respective object record shall prevail.

delimiters	`<repeat.../>`
parent	`<object>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	start position (s-coordinate), overrides the corresponding argument in the original `<object>` record
`length`	double	m	[0,∞[	length of the repeat area
`distance`	double	m	[0,∞[	distance between two instances of the object; If this value is zero, then the object is considered to be a continuous feature like a guard rail, a wall etc.
`tStart`	double	m	]-∞,∞[	lateral offset of object's reference point at s
`tEnd`	double	m	]-∞,∞[	lateral offset of object's reference point at s + length
`widthStart`	double	m	[0,∞[	width of the object at s
`widthEnd`	double	m	[0,∞[	width of the object at s + length
`heightStart`	double	m	]-∞,∞[	height of the object at s
`heightEnd`	double	m	]-∞,∞[	height of the object at s + length
`zOffsetStart`	double	m	]-∞,∞[	z offset of the object at s
`zOffsetEnd`	double	m	]-∞,∞[	z offset of the object at s + length

Illustration:



5.3.8.1.2 Object Outline Record

The default parameters of the object record allow for objects with rectangular and circular footprint to be placed within the database. However, users may need to describe linear features as well as polygonal areas of non-rectangular shape along roads. For this purpose, the outline record may be used. It defines a sequence of corners including height information on the object's extent either in object co-ordinates or relative to the road's reference line (~~mixed definitions may also be used~~). For areas, the points should – preferably – be listed in CCW order.

The outline record must be followed by at least one corner record.

delimiters	`<outline>...</outline>`
parent	`<object>`
instances	`0..1`
attributes	`none`

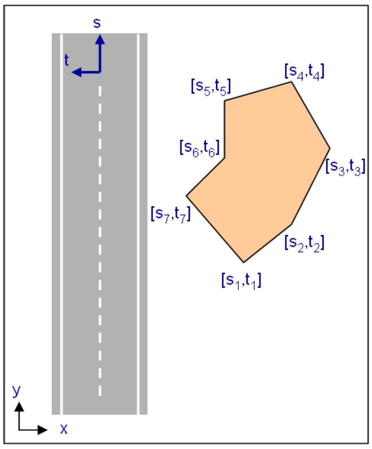
5.3.8.1.2.1 *CornerRoad*

Defines a corner point on the object's outline in road co-ordinates..

delimiters	`<cornerRoad.../>`
parent	`<outline>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	s co-ordinate of the corner
`t`	double	m	]-∞,∞[	t co-ordinate of the corner
`dz`	double	m	]-∞,∞[	delta z of the corner relative to road's reference line
`height`	double	m	]-∞,∞[	height of the object at this corner

Illustration:



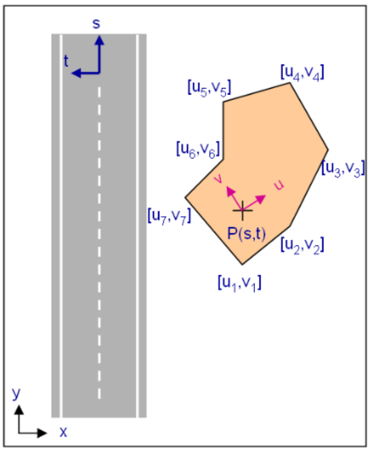
5.3.8.1.2.2 CornerLocal

Defines a corner point on the object's outline relative to the object's pivot point in local u/v co-ordinates. The pivot point and the orientation of the object are given by the `s/t/heading` arguments of the `<object>` entry.

delimiters `<cornerLocal.../>`
parent `<outline>`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`u`	double	m	]-∞,∞[	local u co-ordinate of the corner
`v`	double	m	]-∞,∞[	local v co-ordinate of the corner
`z`	double	m	]-∞,∞[	local z co-ordinate of the corner
`height`	double	m	]-∞,∞[	height of the object at this corner

Illustration:



5.3.8.1.3 Object Material Record

For objects like patches which are within the road surface (and, typically, coplanar to the surface) and which represent a local deviation from the standard road material, a description of the material properties is required. This description supercedes the one provided by the Road Material record and, again, is valid only within the outline of the parent road object.

delimiters `<material.../>`
parent `<object>`
instances `0..1`
attributes

name	***type***	***unit***	***value***	***description***
`surface`	string	-	-	surface material code, depending on application
`friction`	double	-	[0,∞[	friction value
`roughness`	double	-	[0,∞[	roughness (e.g. for sound and motion systems)

5.3.8.1.4 Lane Validity Record

Per default, objects are valid for all lanes pointing into the object's direction. This default validity may be replaced with explicit validity information for an object. The validity record is an optional child record of the object record. Multiple validity records may be defined per object.

delimiters `<validity.../>`
parent `<object>`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`fromLane`	int	-	]-∞,∞[	minimum ID of the lanes for which the object is valid
`toLane`	int	-	]-∞,∞[	maximum ID of the lanes for which the object is valid

NOTE: For single-lane-validity of the object, provide identical values for `fromLane` and `toLane`.

5.3.8.1.5 Parking Space Record

A parking space is a special type of a rectangular object which may carry further information than just its spatial extent. Therefore, details may optionally be added to the object entry in order to define more complex parking space situations.

delimiters	`<parkingSpace>...</parkingSpace>`
parent	`<object>`
instances	`0..1`
attributes	

name	***type***	***unit***	***value***	***description***

`access`	string	-	`all`
			`car`
			`women`
			`handicapped`
			`bus`
			`truck`
				access definitions for the parking space (assuming that "women" and "handicapped" will drive vehicles of type "car" only)

5.3.8.1.6 Parking Space Marking Record

With the marking record, it is possible to define details about the marking on each side of a parking space. In addition, the marking type "curb" is introduced to indicate a hard edge which must not be crossed.

delimiters	`<marking>...</marking>`
parent	`<parkingSpace>`
instances	`0..4`
attributes	

name	***type***	***unit***	***value***	***description***

`side`	string	-	`front`
			`rear`
			`left`
			`right`
				side for which the marking is defined

`type`	string	-	see 6.2 `curb`	type of the marking, additionally "`curb`" is allowed
`width`	double	m	`[0,∞[`	width of the marking
`color`	string	-	see 6.4	color of the marking

5.3.8.2 Object Reference Record

Since the same object may be referred to from several roads, a corresponding record is being provided. This requires, however, that objects which are to be referred to be provided with a unique ID. The object reference record consists of a main record and an optional lane validity record.

delimiters `<objectReference.../>`
parent `<objects>`
instances `0+`
attributes

name	***type***	***type***	***type***	***description***
`s`	double	m	[0,∞[	track position (s-position)
`t`	double	m	]-∞,∞[	lateral position (t-position)
`id`	string	-	-	unique ID of the referred object within the database
`zOffset`	double	m	]-∞,∞[	z offset from track level
`validLength`	double	m	[0,∞[	extent of object's validity along s-axis in (0.0 for point object)

`orientation`	string	-	`+`
			`-`
			`none`
				"+" = valid in positive track direction
				"-" = valid in negative track direction
				"none" = valid in both directions

5.3.8.2.1 Lane Validity Record

see 5.3.8.1.4

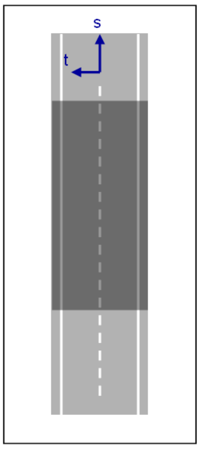
5.3.8.3 Tunnel Record

The tunnel record is – like an object record – applied to the entire cross section of the road within the given range unless a lane validity record with further restrictions is provided as child record.

delimiters	`<tunnel.../>`
parent	`<objects>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	track position (s co-ordinate)
`length`	double	m	[0,∞[	length of the tunnel (s direction)
`name`	string	-	-	name of the tunnel
`id`	string	-	-	unique ID within database
`type`	string	-	see 6.7	type of the tunnel
`lighting`	double	-	[0,1]	degree of artificial tunnel lighting
`daylight`	double	-	[0,1]	degree of daylight intruding the tunnel

Illustration:



5.3.8.3.1 Lane Validity Record

see 5.3.8.1.4.

delimiters	`<validity.../>`
parent	`<tunnel>`

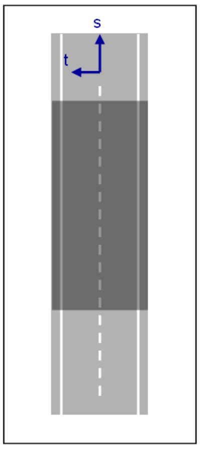
5.3.8.4 Bridge Record

The bridge record is – like an object record – applied to the entire cross section of the road within the given range unless a lane validity record with further restrictions is provided as child record.

delimiters	`<bridge.../>`
parent	`<objects>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	track position (s co-ordinate)
`length`	double	m	[0,∞[	length of the bridge (s direction)
`name`	string	-	-	name of the bridge
`id`	string	-	-	unique ID within database
`type`	string	-	see 6.8	type of the bridge

Illustration:



5.3.8.4.1 Lane Validity Record

see 5.3.8.1.4.

delimiters	`<validity.../>`
parent	`<bridge>`

5.3.9 *Road Signals Record*

The signals record is the container for all signals along a road.

delimiters `<signals>...</signals>`
parent `<road>`
instances `0..1`
attributes `none`

5.3.9.1 Signal Record

The signal record is used to provide information about signs and signals along a road. Signals are signs that can change their state dynamically (e.g. traffic lights). The signal record consists of a main record and an optional lane validity record.

delimiters `<signal.../>`
parent `<signals>`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	track position (s-position)
`t`	double	m	]-∞,∞[	track position (t-position)
`id`	string	-	-	unique ID of the signal within the database
`name`	string	-	-	name of the signal (e.g. bead name in graphics)
`dynamic`	string	-	`yes` `no`	identification whether signal is a dynamic signal (e.g. traffic light)

`orientation`	string	-	`+`
			`–`
			`none`
				"+" = valid in positive track direction
				"-" = valid in negative track direction
				"none" = valid in both directions

`zOffset`	double	m	]-∞,∞[	z offset from track level
`country`	string	-	see 6.10	country code of the signal
`type`	int	-	-1 see 6.11	type number according to country code or -1
`subtype`	int	-	-	subtype number according to country code or -1
`value`	double	-	-	value of the signal (e.g. speed, mass – depending on type)
`unit`	string	-	see 2.2	unit of the attribute `value` (optional)
`height`	double	m	[0,∞[	height of the signal
`width`	double	m	[0,∞[	width of the signal
`text`	string	-	string	additional text associated with the signal (e.g. text on city limit "Stadt\nBadAibling")
`hOffset`	double	rad	]-∞,∞[	heading offset of the signal (relative to orientation)
`pitch`	double	rad	]-∞,∞[	pitch angle of the signal
`roll`	double	rad	]-∞,∞[	roll angle of the signal

5.3.9.1.1 Lane Validity Record

Per default, signals are valid for all lanes pointing into the signal's direction. This default validity may be replaced with explicit validity information for a signal. The validity record is an optional child record of the signal record. Multiple validity records may be defined per signal.

delimiters	`<validity.../>`
parent	`<signal>`
instances	0+
attributes	

name	***type***	***unit***	***value***	***description***
`fromLane`	int	-	]-∞,∞[	minimum ID of the lanes for which the object is valid
`toLane`	int	-	]-∞,∞[	maximum ID of the lanes for which the object is valid

NOTE: For single-lane-validity of the signal, provide identical values for `fromLane` and `toLane`.

5.3.9.1.2 Signal Dependency Record

The signal dependency record provides signals with a means to control other signals. Signs can e.g. restrict other signs for various types of vehicles, warning lights can be turned on when a traffic light goes red etc. The signal dependency record is an optional child record of the signal record. A signal may have multiple dependency records.

delimiters	`<dependency.../>`
parent	`<signal>`
instances	0+
attributes	

name	***type***	***unit***	***value***	***description***
`id`	string	-	-	ID of the controlled signal
`type`	string	-	-	type of the dependency, depending on the application

5.3.9.2 Signal Reference Record

Depending on the way roads (especially in junctions) are laid out for different applications, it may be necessary to refer to the same (i.e. the identical) sign from multiple roads. In order to prevent inconsistencies by multiply defining an entire signal entry, the user only needs to define the complete signal entry once and can refer to this complete record by means of the signal reference record.

This requires, however, that signals which are to be referred to be provided with a unique ID.
The signal reference record consists of a main record and an optional lane validity record.

delimiters `<signalReference.../>`
parent `<signals>`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`s`	double	m	[0,∞[	track position [m] (s-position)
`t`	double	m	]-∞,∞[	track position [m] (t-position)
`id`	string	-	-	unique ID of the referenced signal within the database

`orientation`	string	-	`+`
			`-`
			`none`
				“+” = valid in positive track direction
				“-“ = valid in negative track direction
				“none” = valid in both directions

5.3.9.2.1 Lane Validity Record

see 5.3.9.1.1.

5.3.10 Surface

The surface record is the container for all surface descriptions which shall be applied along a road.

delimiters	`<surface>...</surface>`
parent	`<road>`
instances	`0..1`
attributes	`none`

5.3.10.1 Curved Regular Grid Record

The interface to a Curved Regular Grid (CRG) surface description file is defined as arguments to the `<CRG>` tag and as an include operation between the opening and closing `<CRG>` tags.

delimiters	`<CRG>...</CRG>`
parent	`<surface>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`file`	string	-	-	name of the file containing the CRG data
`sStart`	double	m	[0,∞[	start of the application of CRG data (s-position)
`sEnd`	double	m	[0,∞[	end of the application of CRG (s-position)
`orientation`	string	-	`same` `opposite`	orientation of the CRG data set relative to the underlying road

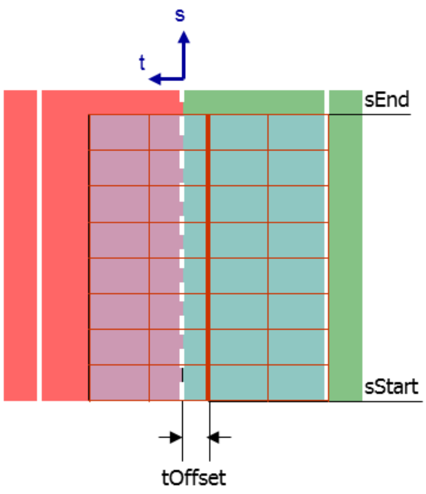
`mode`	string	-	`attached`
			`attached0`
			`genuine`
				application mode for the surface data

`purpose`	string	-	`elevation` `friction`	optional, physical purpose of the data contained in the CRG file; if the attribute is missing, data will be interpreted as elevation data.
`sOffset`	double	m	]-∞,∞[	s-offset between CRG center line and reference line of the road (optional, default = 0.0)
`tOffset`	double	m	]-∞,∞[	t-offset between CRG center line and reference line of the road (optional, default = 0.0)
`zOffset`	double	m	]-∞,∞[	z offset between CRG center line and reference line of the road (optional, default = 0.0)
`zScale`	double	-	]-∞,∞[	z scale factor for the surface description (optional, default = 1.0)
`hOffset`	double	rad	]-∞,∞[	heading offset between CRG center line and reference line of the road (required for mode `genuine` only, optional, default = 0.0)

As the name indicates, CRG data is organized in a regular grid which is laid out along a reference line (comparable to an OpenDRIVE road's reference line). At each grid position, it contains the absolute elevation measured along a real road and some additional data which allows for the computation of the delta elevation relative to the reference line.

The key to combining OpenDRIVE and CRG data is to define a correlation between the two reference lines and a rule for using the elevation data of both descriptions.

CRG data may be offset from the OpenDRIVE road's reference line (see `tOffset`) and it may be oriented in the same or opposite direction as the layout direction of the road (see `orientation`).



The CRG data may be applied to a given OpenDRIVE road in two modes:

Mode `attached`:

The reference line of the CRG data set is replaced with the OpenDRIVE road's reference line, taking into account the `tOffset` and the `sOffset` parameters

The CRG local elevation values (calculated by evaluating the CRG grid and applying `zOffset` and `zScale`) will be added to the surface elevation data of the OpenDRIVE road (as derived from the combination of elevation, super-elevation and crossfall).

With this mode, the surface information relative to the original CRG data's reference line is transferred from an arbitrary CRG road to an OpenDRIVE road without having to make sure that the overall geometries of the road match. The original position, heading, curvature, elevation and superelevation of the CRG road are disregarded. The CRG grid is evaluated along the OpenDRIVE reference line instead of the CRG reference line.

$$\begin{pmatrix} u \\ v \end{pmatrix}_{CRG} = \begin{pmatrix} s - s_{Offset} \\ t - t_{Offset} \end{pmatrix}_{OpenDrive}$$

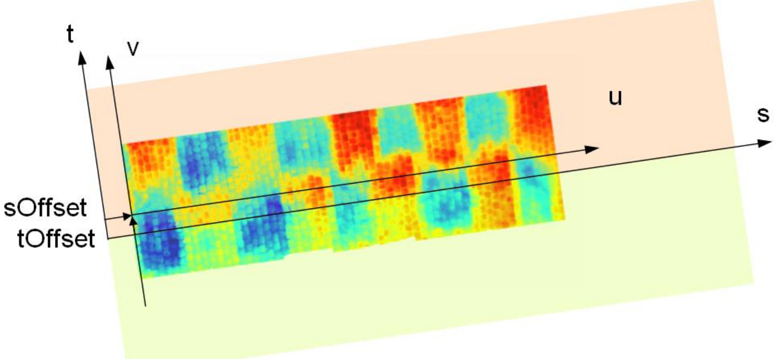


Image courtesy of Daimler AG

Mode `attached0`:

This mode is basically the same as the `attached` mode, with the only exception that only the CRG data's elevation value is taken into account (i.e. the OpenDRIVE elevation is set to zero)

Mode `genuine`:

The start point of the CRG data set's reference line is positioned relative to the point on the OpenDRIVE road's reference line at the position defined by `sStart, sOffset` and `tOffset`.

By providing offset values for the longitudinal (`sOffset`) and lateral (`tOffset`) displacement, the heading (`hOffset`) and the elevation (`zOffset`), the correlation between the two description's reference lines is clear.

In genuine mode, the CRG data will completely replace the OpenDRIVE elevation data, i.e. the absolute elevation of a given point of the road surface is directly computed from the CRG data (think of it as combining CRG and OpenDRIVE data with the OpenDRIVE elevation, super-elevation and crossfall all being zero).

When using this method, it must of course be made sure that the geometry of the CRG data matches - within certain tolerance - the geometry of the underlying OpenDRIVE road.

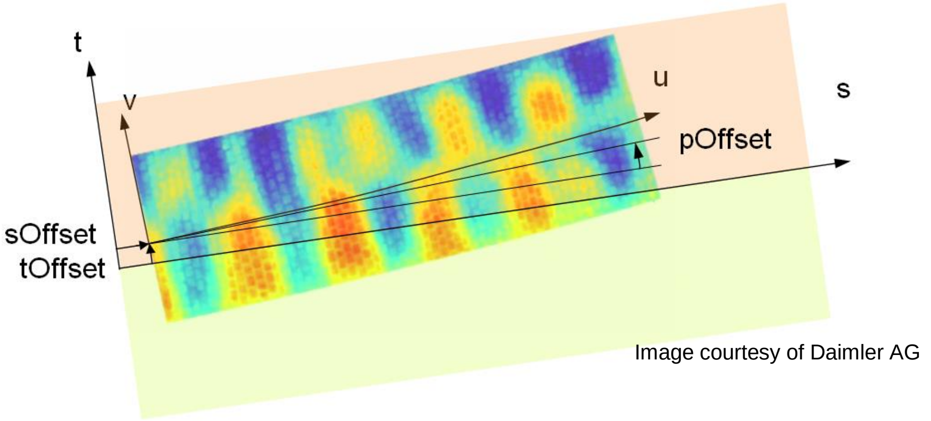


Image courtesy of Daimler AG

Since CRG data may only cover parts of a road's surface, it must be made sure that outside the valid CRG area, the elevation information derived from OpenDRIVE data can still be used.

Example:

```
<surface>
    <CRG file="fancyData.crg" sStart="0.0" sEnd="100.0"
         orientation=”same” mode=”relative”
         tOffset=”0.0”>
    </CRG>
</surface>
```

5.3.11 Railroad Elements

The available set of railroad records below the road level is currently limited to the definition of switches. All other entries shall, for the moment, be covered with the existing records (e.g. track definition by `<road>`, signal definition by `<signal>` etc.). It shall be noted again, that railroad specific elements are defined on the background of streetcar applications.

The railroad record is the container for all railroad definitions which shall be applied along a road.

delimiters	`<railroad>...</railroad>`
parent	`<road>`
instances	`0..1`
attributes	`none`

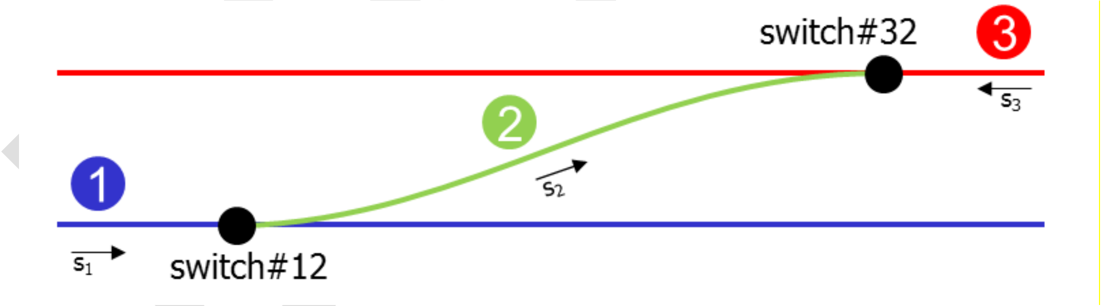
5.3.11.1 Railroad Switches

delimiters	`<switch>...</switch>`
parent	`<railroad>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	unique name of the switch
`id`	string	-	-	unique id of the switch (preferably an integer number, uint32_t)

`position`	string	-	`dynamic`
			`straight`
			`turn`
				either a switch can be operated (`dynamic`) or it is in a static position

Ilustration:



5.3.11.1.1 Main Track

delimiters	`<mainTrack>...</mainTrack>`
parent	`<switch>`
instances	`1`
attributes	

name	***type***	***unit***	***value***	***description***
`id`	string	-	-	unique ID of the main track (i.e. `<road>` entry)
`s`	double	m	[0,∞[	s position of the switch (i.e. the point where main track and side track meet)
`dir`	string	-	+ −	driving direction on the main track for entering the side track via the switch

5.3.11.1.2 Side Track

delimiters	`<sideTrack>...</sideTrack>`
parent	`<switch>`
instances	`1`
attributes	

name	***type***	***unit***	***value***	***description***
`id`	string	-	-	unique ID of the side track (i.e. `<road>` entry)
`s`	double	m	[0,∞[	s position of the switch on the side track
`dir`	string	-	+ −	driving direction on the side track for after entering it via the switch

5.3.11.1.3 Partner Switch

This entry may be used to indicate which switch will lead out of a side track after it has been entered. This entry is optional and may be used for easier navigation through railroad networks. In the example above, the switches 12 and 32 would be considered partners.

delimiters	`<partner>...</partner>`
parent	`<switch>`
instances	`0..1`
attributes	

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	unique name of the partner switch
`id`	string	-	-	unique id of the partner switch

5.4 Controller Record

A controller provides consistent states for a group of signals. This may be a set of signals within a junction or a set of dynamic speed restrictions on a motorway. The entire record consists of a header followed by a number of dependency records.

delimiters `<controller.../>`
parent `<OpenDRIVE>`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`id`	string	-	-	unique ID within database
`name`	string	-	-	name of the controller
`sequence`	uint32_t	-	[0,∞[	sequence number (priority) of this controller with respect to other controllers of same logical level

5.4.1 Control Entry Record

The control entry record provides information about a single signal controlled by the corresponding controller. This record is a child record of the controller record.

delimiters `<control.../>`
parent `<controller>`
instances `1+`
attributes

name	***type***	***unit***	***value***	***description***
`signalId`	string	-	-	ID of the controlled signal
`type`	string	-	-	type of control

5.5 Junction Record

When a road can be linked to more than one successor (or predecessor, depending on the orientation), a junction record is required. It contains the information about all possible connections between roads meeting at a physical junction.

For junctions, two types of roads must be distinguished:

- standard roads
- paths

`Standard roads` are in-coming and out-going roads of the junction. Usually, they don't require any special treatment except for the fact that they end or begin at a junction.

`Paths` are roads within a junction. For each connection leading from one standard road to another, a path is defined. Paths may link only single lanes or a set of lanes at once. Paths must only contain lanes of one side (left or right). However, within the database, paths having only left lanes and paths having only right lanes may both be contained. The geometric and logic description of paths complies with the rules defined for standard roads (i.e. they contain lanes, elevation entries etc.).

Due to the fact that paths cannot have lanes of both directions, junctions only provide information linking incoming roads to paths. This is the only ambiguous part of a connection. The link between a path and the corresponding outgoing road is clear from the ROAD LINK record of the path (see 5.3.2).

The junction record is split into a header record and a series of link records.

delimiters	`<junction>...</junction>`
parent	`<OpenDRIVE>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	name of the junction
`id`	string	-	-	unique ID within database

5.5.1 Connection Record

The connection record provides information about a single connection within a junction. It is a child of the junction header record.

delimiters	`<connection>...</connection>`
parent	`<junction>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`id`	string	-	-	unique ID within the junction
`incomingRoad`	string	-	-	ID of the incoming road
`connectingRoad`	string	-	-	ID of the connecting path
`contactPoint`	string	-	*`start`* *`end`*	contact point on the connecting road

5.5.1.1 Junction Lane Link Record

The junction lane link record provides information about the lanes which are linked between incoming road and connecting road. This record may be omitted if all incoming lanes are linked to lanes with identical IDs on the connecting road. However, it is strongly recommended to provide this record.

delimiters	`<laneLink.../>`
parent	`<connection>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`from`	int	-	]-∞,∞[	ID of the incoming lane
`to`	int	-	]-∞,∞[	ID of the connecting lane

5.5.2 Junction Priority Record

The junction priority record provides information about the priority of a connecting road over another connecting road. It is only required if priorities cannot be derived from signs or signals in a junction or on tracks leading to a junction.

delimiters	`<priority.../>`
parent	`<junction>`
instances	`0+`
attributes	

name	type	unit	value	description
`high`	string	-	-	ID of the prioritized connecting road
`low`	string	-	-	ID of the connecting road with lower priority

5.5.3 Junction Controller Record

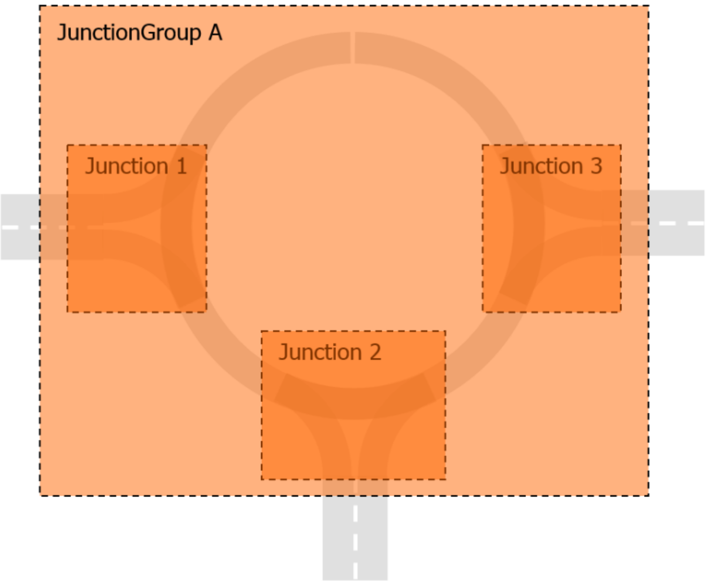
Junction controller records list the controllers which are used for the management of a junction.

delimiters	`<controller.../>`
parent	`<junction>`
instances	`0+`
attributes	

name	type	unit	value	description
`id`	string	-	-	ID of the controller
`type`	string	-	-	type of control for this junction
`sequence`	uint32_t	-	[0,∞[	sequence number (priority) of this controller with respect to other controllers in the same junction

5.6 Junction Group Record

In some cases (e.g. roundabout) it might be good to group junctions so that the traffic simulation software can execute calculations more efficiently.



The junction group record is split into a header record and a series of member records.

delimiters	`<junctionGroup>...</junctionGroup>`
parent	`<OpenDRIVE>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	name of the junction group
`id`	string	-	-	unique ID within database
`type`	string	-	`roundabout` `unknown`	type of this grouping

5.6.1 Junction Reference Record

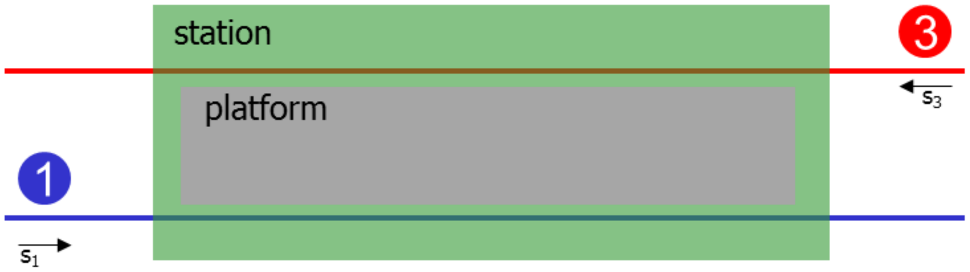
Junction group members are nothing but references to existing junction records.

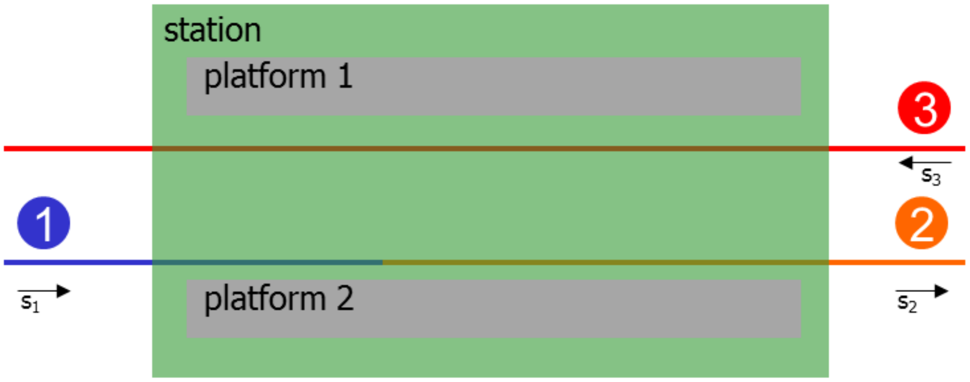
delimiters	`<junctionReference.../>`
parent	`<junctionGroup>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`junction`	string	-	-	ID of the junction

5.7 Stations

For tram and railroad applications, the definition of stations must be feasible. Stations may refer to multiple tracks and are, therefore, defined on the same level as junctions. The physical extent of a station is defined by the combined physical extent of all platforms within the station. For pure automotive environments, also bus stations may be defined using the same elements.





The station record is defined as follows.

delimiters	`<station>...</station>`
parent	`<OpenDRIVE>`
instances	`0+`
attributes	

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	name of the station
`id`	string	-	-	unique ID within database

`type`	string	-	`small`
			`medium`
			`large`
				type of the station

5.7.1 Platform Record

Each station entry must contain at least one platform entry. Each platform entry must contain at least one reference to a valid track segment.

delimiters	`<platform>...</platform>`
parent	`<station>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`name`	string	-	-	name of the platform
`id`	string	-	-	unique ID within database

5.7.1.1 Segment Record

Each platform entry is valid on one or more track segments. These have to be specified using the following record.

delimiters	`<segment.../>`
parent	`<platform>`
instances	`1+`
attributes	

name	***type***	***unit***	***value***	***description***
`roadId`	string	-	-	unique ID of the road (track) which accompanies the platform
`sStart`	double	m	[0,∞[	minimum s-value on road where platform is adjacent to it
`sEnd`	double	m	[0,∞[	maximum s-value on road where platform is adjacent to it
`side`	string	-	`left` `right`	side of track where platform is situated when going from `sStart` to `sEnd`.

5.8 Ancillary Data

Ancillary data for a bead may be defined by generating an entry that directly follows the corresponding bead. It is designed to contain user-defined data that is required for specific applications or is not yet covered by the OpenDRIVE® standard.

delimiters `<userData.../>`
parent `any`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`code`	string	-	-	code for the user data (application specific)
`value`	string	-	-	user data as string (e.g. hexdump)

5.9 Include Tag

The inclusion of another file can be triggered with an include tag at any location.

delimiters `<include.../>`
parent `any`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`file`	string	-	-	location of the file which is to be included

5.10 Alternative Layouts (Sets)

Sets indicate that more than one setup of the properties enclosed by the set's opening and closing tags are available and that the application may choose one of these setups (the activation of none or more than one setup at a given time is not supported). Sets may be used e.g. for different road mark setups, different signaling etc.

delimiters `<set>...</set>`
parent `any`
instances `0+`
attributes

name	***type***	***unit***	***value***	***description***
`id`	string	-	-	unique ID within the database

5.10.1 Layout Instance

Each set may contain one or more alternative setups of the enclosed property. In order to identify a specific setup, it must be enclosed with instance tags.

delimiters `<instance>...</instance>`
parent `<set>`
instances `1+`
attributes

name	***type***	***unit***	***value***	***description***
`id`	string	-	-	unique ID within the database

Example:

```
<road ...>
    <set id="roadWorks">
        <instance id="regular">
            <lanes>
            :
            </lanes>
        </instance>
        <instance id="construction">
            <lanes>
            :
            </lanes>
        </instance>
    </set>
</road>
```

6 Constants

6.1 Road Type Information

The known keywords for the road type information are:

```
unknown
rural
motorway
town
lowSpeed
pedestrian
```

NOTE: In Germany, `lowSpeed` is equivalent to a 30km/h zone

6.2 Road Mark Type Information

The known keywords for the simplified road mark type information are:

```
none
solid
broken
solid solid  (for double solid line)
solid broken  (from inside to outside, exception: center lane - from left to right)
broken solid  (from inside to outside, exception: center lane - from left to right)
botts dots
grass (meaning a grass edge)
curb
```

6.3 Road Mark Weight Information

The known keywords for the simplified road mark weight information are:

```
standard
bold
```

6.4 Road Mark Color Information

The known keywords for the road mark color information are:

```
standard (equivalent to "white")
blue
green
red
white
yellow
```

6.5 Lane Type Information

The known keywords for the lane type information are:

```
none
driving
stop
shoulder
biking
sidewalk
border
restricted
parking
mwyEntry
mwyExit
bidirectional (e.g. for center turn lanes)
median
special1
special2
special3
roadWorks
tram
rail
```

6.6 Object Types

The known keywords for the object type information are:

```
none (i.e. unknown)
obstacle (for anything not further categorized)
car
truck
van
bus
trailer
bike
motorbike
tram
train
pedestrian
pole
tree
vegetation
barrier
building
parkingSpace
wind
patch
```

6.7 Tunnel Types

The known keywords for the tunnel type information are:

```
standard
underpass (i.e. sides are open for daylight)
```

6.8 Bridge Types

The known keywords for the bridge type information are:

```
concrete
steel
brick
wood
```

6.9 Access Restriction Types

The known keywords for the restriction information are:

```
simulator
autonomous traffic
pedestrian
none
```

6.10 Signal Country Codes

The known keywords for the signal country codes are:

```
OpenDRIVE
Austria
Brazil
China
France
Germany
Italy
Switzerland
USA
```

6.11 Signal Types

For the country codes “Germany” and “OpenDRIVE”, the following signal types shall be defined in addition to the numbers given by the corresponding rule books:

traffic light	1,000,001
pedestrian traffic light	1,000,002

Others may be added in future version of this specification.

7 Further Explanations and Examples

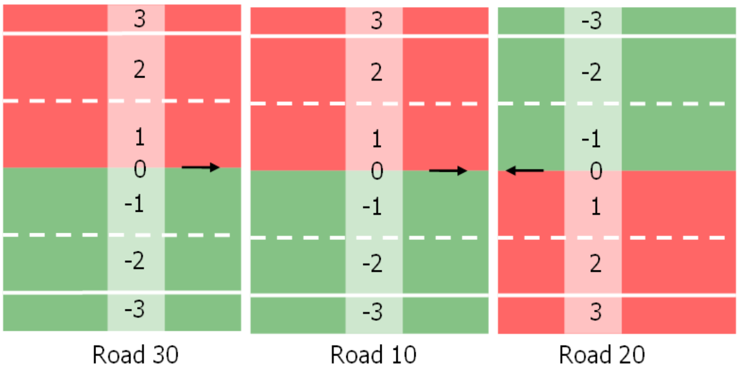
Important Note: All of the following examples do not provide complete OpenDRIVE data sets. Instead, they are reduced to the contents necessary for explaining the respective issues.

7.1 Road and Lane Linkage

7.1.1 Direct Link of Roads and Lanes

Two roads which connect directly with each other, can be handled without any junction entries. Instead, the link information will be provided within the road tags.

Let's assume the following situation:



Road 10 is directly linked to road 20 and road 30. The cross sections of all roads are identical, however the directions do not match between road 10 and 20.

The following database fragment provides the link information on two levels. First, the direct link between the roads is indicated in the <link>-tag which follows the <road>-tag.

```
<road name="" length="1.0000000000000000e+00" id="10" junction="-1">
   <link>
      <predecessor elementType="road" elementId="30" contactPoint="end" />
      <successor elementType="road" elementId="20" contactPoint="end" />
   </link>
```

Road 30 is the predecessor of road 10 and road 10 connects to the end of road 30
Road 20 is the successor of road 10 and road 10 connects to the end of road 20

Now that the basic road connections are known, one can provide additional information about the lane links. If a road has multiple lane sections then the first and last lane section are of interest for the transition from one road's lane to another road's lane. Here, let's assume, our roads have only one lane section each.

So, the entries for road 10 look like the following:

```
<lanes>
    <laneSection s="0.0000000000000000e+00">
        <left>
            <lane id="3" type="border" level= "0">
                <link>
                    <predecessor id="3"/>
                    <successor id="-3"/>
                </link>
            </lane>
            <lane id="2" type="driving" level= "0">
                <link>
                    <predecessor id="2"/>
                    <successor id="-2"/>
                </link>
            </lane>
            <lane id="1" type="driving" level= "0">
                <link>
                    <predecessor id="1"/>
                    <successor id="-1"/>
                </link>
            </lane>
        </left>
        <center>
            <lane id="0" type="border" level= "0">
                <link>
                </link>
            </lane>
        </center>
        <right>
           <lane id="-1" type="driving" level= "0">
                <link>
                    <predecessor id="-1"/>
                    <successor id="1"/>
                </link>
            </lane>
            <lane id="-2" type="driving" level= "0">
                <link>
                    <predecessor id="-2"/>
                    <successor id="2"/>
                </link>
            </lane>
            <lane id="-3" type="border" level= "0">
                <link>
                    <predecessor id="-3"/>
                    <successor id="3"/>
                </link>
            </lane>
        </right>
    </laneSection>
</lanes>
```

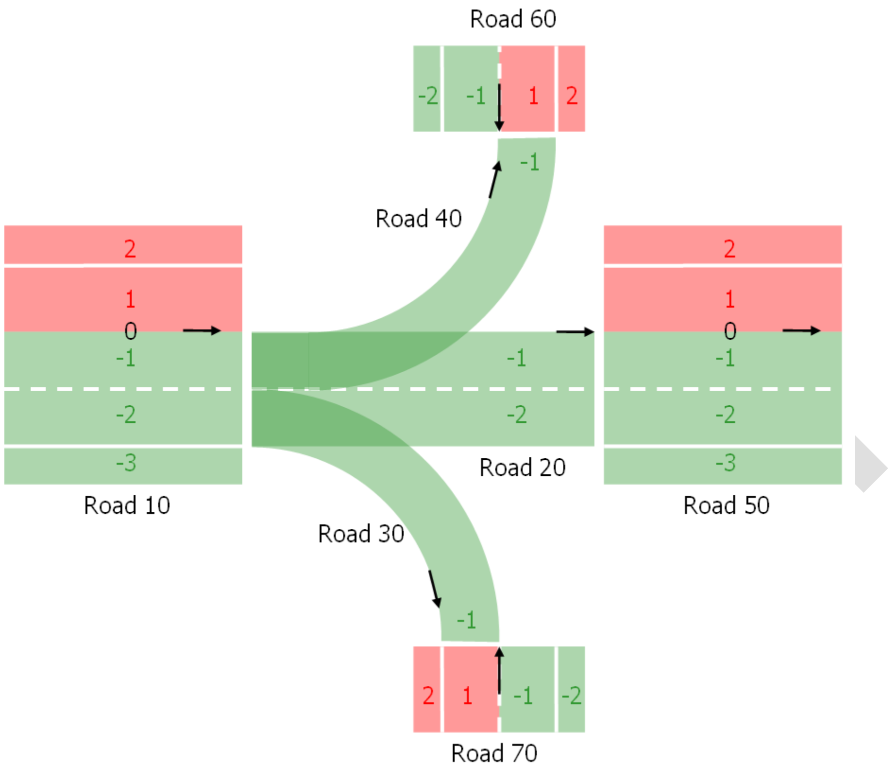
The predecessor of lane 3 on road 10 is lane 3 on road 30.

The successor of lane 3 on road 10 is lane -3 on road 20.

The same scheme applies to all other lanes of the cross section.

7.1.2 Links using Junctions

If the successor or predecessor of a road becomes ambiguous, then a junction is required. Let's take a look at the following example:



Start on the left side: Road 10 links to a junction. The depicted connections (which do not represent the entire connection matrix) allow for the following ways to navigate the junction:

Road 10, lane -1	via	road 20, lane -1	to	road 50, lane -1
	or via	road 40, lane -1	to	road 60, lane 1
Road 10, lane -2	via	road 20, lane -2	to	road 50, lane -2
	or via	road 30, lane -1	to	road 70, lane 1

So, we have two incoming lanes and four possible destinations.

The links between the connecting roads 20, 30, and 40 and their respective successors and predecessors (note that all connecting roads in the example only have right lanes) are clear and can be handled like the standard linking described in the previous chapter.

Thus, for road 30, the link information can be provided in the following way:

```
<road name="" length="1.0000000000000000e+00" id="30" junction="-1">
   <link>
       <predecessor elementType="road" elementId="10" contactPoint="end" />
       <successor elementType="road" elementId="70" contactPoint="end" />
   </link>
   <lanes>
       <laneSection s="0.0000000000000000e+00">
           <left>
           </left>
           <center>
               <lane id="0" type="border" level= "0">
                   <link>
                   </link>
               </lane>
           </center>
           <right>
              <lane id="-1" type="driving" level= "0">
                   <link>
                       <predecessor id="-2"/>
                       <successor id="1"/>
                  </link>
               </lane>
          </right>
       </laneSection>
   </lanes>
```

For road 10, the successor is now a junction instead of a road. This is indicated in the link information following the <road> tag:

```
<road name="" length="2.0000000000001661e+00" id="10" junction="-1">
    <link>
        <predecessor elementType="road" elementId="99" contactPoint="end" />
        <successor elementType="junction" elementId="25" />
    </link>
```

```
<lanes>
    <laneSection s="0.0000000000000000e+00">
        <left>
            <lane id="2" type="driving" level= "0">
                <link>
                    <predecessor id="2"/>
                </link>
            </lane>
            <lane id="1" type="driving" level= "0">
                <link>
                    <predecessor id="1"/>
                </link>
            </lane>
        </left>
        <center>
            <lane id="0" type="border" level= "0">
                <link>
                </link>
            </lane>
        </center>
        <right>
           <lane id="-1" type="driving" level= "0">
                <link>
                    <predecessor id="-1"/>
                </link>
            </lane>
            <lane id="-2" type="driving" level= "0">
                <link>
                    <predecessor id="-2"/>
                </link>
            </lane>
            <lane id="-3" type="border" level= "0">
                <link>
                    <predecessor id="-3"/>
                </link>
            </lane>
        </right>
    </laneSection>
</lanes>
```

The sample above assumes road 10 has a predecessor no. 99 (not depicted above, so let's assume it is of the same cross-section as road 10) and a successor which is junction no. 25. The junction itself is not shown since it is just a logical container for further connection information (see below)

The presence of the junction is also considered in the lane links of road 10. Only the links to the predecessor are clear. The links to the successors are omitted. This can also be seen in the code sample on the left.

Now for the corresponding junction matrix which closes the gap between the incoming road 10 and all possible destinations (roads 50, 60 and 70). Again, the information is provided on two levels, first the road level, then the lane level. Look at the following code fragment:

```
<junction name="" id="25">
    <connection id="0" incomingRoad="10" connectingRoad="20" contactPoint="start">
        <laneLink from="-1" to="-1"/>
        <laneLink from="-2" to="-2"/>
    </connection>
    <connection id="1" incomingRoad="10" connectingRoad="30" contactPoint="start">
        <laneLink from="-2" to="-1"/>
    </connection>
    <connection id="2" incomingRoad="10" connectingRoad="40" contactPoint="start">
        <laneLink from="-1" to="-1"/>
    </connection>
</junction>
```

As can be seen, the connection from road 10 to road 20 may be entered on two different lanes, whereas the other connections may be entered on one lane only.

As state above, the links between the connecting roads and their successors is, again, clear and can therefore be provided as standard <link> information below the <road> tag and the respective <lane> tags.

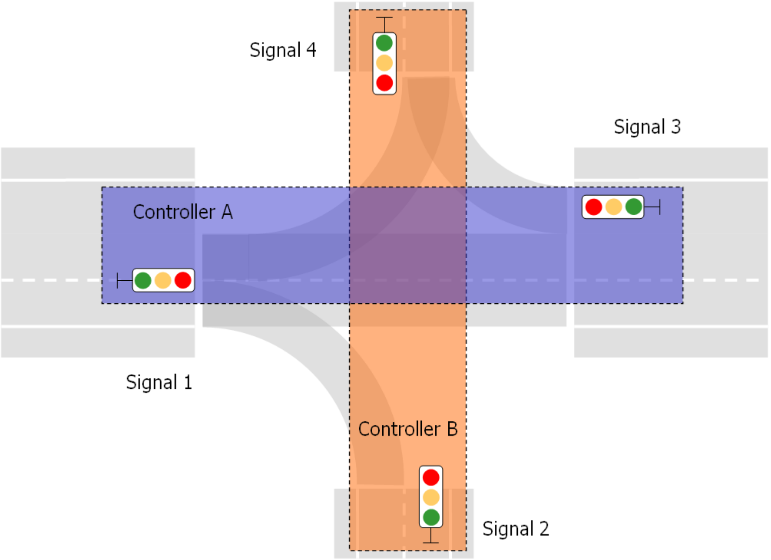
7.2 Junction Control

The designer of a road network does not only provide lane and road linkage information but also details of the signalization. It is assumed to be his (or her) task to group signals in a fashion that they can be controlled consistently from the scenario simulation which reads and interprets the OpenDRIVE file.

Signal control is split into two levels which allow for the grouping of

- various signals under a single controller and for
- various controllers within one junction

The following example shall provide a control scheme of a "standard" crossing. It represents one possible solution only which is, of course, non-binding.



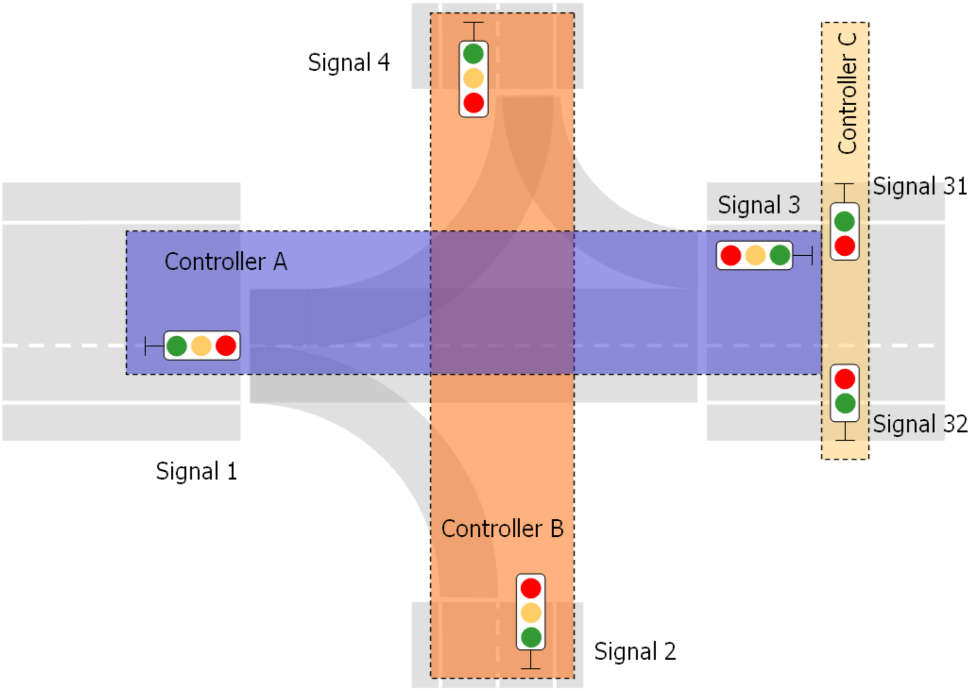
It is recommended to assign synchronous signals of same type to identical controllers instead of having one controller for each signal. So, in the example above, both through-directions will be controlled by individual controllers.

In the OpenDRIVE file, the controllers would be defined in the following way:

```
<OpenDRIVE>
    <controller id="Controller A" type="">
        <control signalId="Signal 1" type=""/>
        <control signalId="Signal 3" type=""/>
    </controller>
    <controller id="Controller B" type="">
        <control signalId="Signal 2" type=""/>
        <control signalId="Signal 4" type=""/>
    </controller>
</OpenDRIVE>
```

Please note that the type of control is not yet actually used for controller or control entry.

If, for example, pedestrian lights should be added to the above junction, it is recommended to group them under additional controllers. The following image illustrates this situation:



In this example, the signals 31 and 32 are controlled by an additional controller "Controller C". If corresponding pedestrian signals existed at the location of "Signal 1", they could also be controlled by "Controller C".

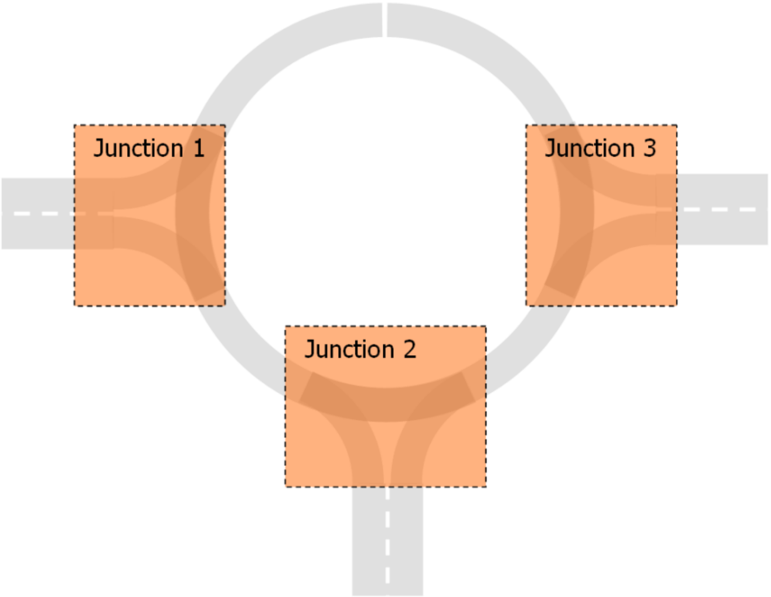
Finally, all controllers responsible for a junction may be grouped within the <junction> tag. Again, let's assume, we are talking about junction no. 25 in the example above.

The junction entry would look similar to the following one (not all connections and controls are given):

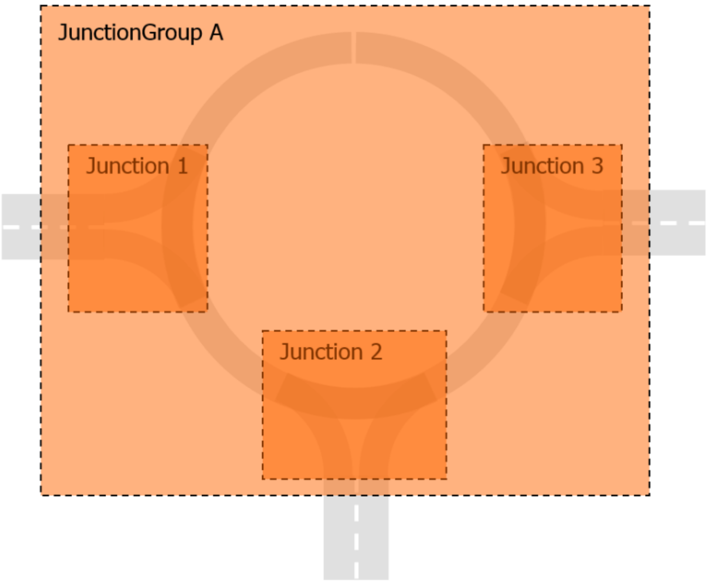
```
<OpenDRIVE>
    <junction name="" id="25">
        <connection id="0" incomingRoad="10" connectingRoad="20" contactPoint="start">
            <laneLink from="-1" to="-1"/>
            <laneLink from="-2" to="-2"/>
        </connection>
        <connection id="1" incomingRoad="10" connectingRoad="30" contactPoint="start">
            <laneLink from="-2" to="-1"/>
        </connection>
        <connection id="2" incomingRoad="10" connectingRoad="40" contactPoint="start">
            <laneLink from="-1" to="-1"/>
        </connection>
        <controller id="Controller A" type=""/>
        <controller id="Controller B" type=""/>
        <controller id="Controller C" type=""/>
    </junction>
</OpenDRIVE>
```

7.3 Roundabouts

Roundabouts are to be considered as collections of junctions with each junction being constructed according to the rules explained above. Also for the signaling, the rules laid out before apply.



All junctions belonging to a single roundabout shall be referenced under a junction group:


JunctionGroup A
Junction 1
Junction 3
Junction 2