Technische Universität München - Lehrstuhl für Verkehrstechnik
Univ.-Prof. Dr.-Ing. Fritz Busch
Arcisstraße 21 80333 München, www.vt.bgu.tum.de

MASTER'S THESIS

Extension of the Geospatial Data Abstraction Library (GDAL/OGR) for OpenDRIVE Support in GIS Applications for Visualisation and Data Accumulation for Driving Simulators

Author:

Ing.-Tel. Ana María Orozco Idrobo

Supervision:

Dipl.-Ing. Martin Margreiter (TUM)

Mentoring:

Dipl.-Geoinf. Michael Scholz

Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)

German Aerospace Center

Institute of Transportation Systems, Automotive

Date of Submission: 2015.12.11

Deutsches Zentrum
für Luft- und Raumfahrt
German Aerospace Center

Institute of Transportation Systems

Prof. Dr. Frank Köster
Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)
Institut für Verkehrssystemtechnik, Braunschweig

Dipl.-Geoinf. Michael Scholz
Institut für Verkehrssystemtechnik, Automotive

Abstract

Intelligen Transportation Systems (ITS) are envisioned to improve road safety, traffic efficiency, sustainable transport and information services, through multi-modal means of transportation and its integration across an Information and Communications (ICT) platform. In this sense, many location-based services demand the support of specific source of georeferenced data and its modeling. Due to the nature of the spatial data, *Geographic Information Systems* (GIS) and techniques are required for the analysis and manipulation of geodata. Moreover, given the intrinsic relationship between ITS and the geo-sciences, the terms GIS-T or Geo-ITS have been coined in the last years.

Major challenges regarding the transport networks generation include the heterogeneity of the data and the diversity of the sources, as well as, different content and formats. In addition to this, the geodata require accuracy and precision, particularly for simulation purposes and real-world implementations. In this context, the spatial data are key components for the simulation and the spatial processing engines in the ITS sphere. Nevertheless, one problem is the few available geotools for management and visualization of the data, for one of the most used simulation formats: the OpenDRIVE standard.

OpenDRIVE is considered the standard de-facto for the driving simulation community, however, few management tools have been provided to facilitate the manipulation and visualization of these data. For this reason, this master thesis focuses on the conceptualization, modeling and implementation of the *XODR-Driver*; a geo-processing toolbox for the OpenDRIVE data format. The *XODR-Driver* provides the functionality to translate the OpenDRIVE files to the standard GIS vectorial representation. *XODR-Driver* also enables the general GIS framework and its inherit vectorial operations for the OpenDRIVE format.

The contribution of this work is the *OpenDRIVE XODR-Driver*, a functional software module, with a modular architecture which implements the mathematical models for the road layout descriptions of the reference line or track.

Contents

Introduction

In a broad sense the *Intelligent Transportation Systems* (ITS) aim four major goals: safety, efficient, comfortable and environmental friendly means of transport. These objectives can be achieved through the integration and the interoperability of the transportation systems, the infrastructure, the technological platforms and the information systems.

In this context, the management of the spatial information plays a relevant role for the ITS applications. Due to the nature of the geographic data, an special treatment, muss be given to the geo-information. In this sense, the *geodata* and the transportation systems are intrinsically linked; therefore, rise the need of handling the spatial data to operate over the datasets and to performs the inherited function of the *Geographic Information Systems* (GIS) domain, such as data acquisition, modeling, integration to the visualization and analysis.

On the field of Intelligent Transportation systems, the applications based on *geo-referenced* data are exponentially growing [Miller and Shaw, 2001]: from the traffic management tools and control systems, traffic and driving simulator to the individual mobile applications in our daily life and the future autonomous driving technologies.

The applications of the GIS on the transport field are innumerable, because the GIS not only respond to the *where* question, but explain the phenomena related to specific locations. In this scope, the world of GIS demands complex analysis operations and visualization techniques over datasets. For many years the commercial and the open formats are part of the vast number of digital data formats: typically raster or vector. In that order of ideas, the OpenDRIVE standard provides the format data for road networks representation. As a matter of fact, OpenDRIVE is consider as the de-facto standard for the driving simulation industry.

Furthermore, transportation systems like driving assisted systems, autonomous driving depends on models and simulations of the vehicle dynamics, the traffic flow and the driving behavior for testing and calibrating purposes. In this respect, these ITS systems demand advanced techniques of mapping, merging, 3D visualization and images processing, in order to offer a realistic simulation in urban scenarios.

In addition, OpenDRIVE offers a broad range of applications in the transportation area: for city planners, for public transport authorities, for traffic and control engineers, etc. OpenDRIVE provides the flexibility to manage and process geodata, in combination with other sources and content. Some examples are the navigation and routing systems, the optimization of the transport networks, planning activities, and so forth. Nevertheless, one problem is advised with the OpenDRIVE format: the data representation of the road networks are given by a non-standard format.

Thus, this master thesis addresses to model, design and implement the vectorial format of OpenDRIVE; by means of a software converter or ***XODR-Driver*** for general use on the ITS and GIS spheres. This work comprehends the mathematical conceptualization of the reference lines of the roads and the elements along. Also, the software design and characterization of the driver modules and the development and integration of the XODR-Driver in the *Geospatial Data Abstraction Library* (GDAL). Therefore, three parts are distinguished on this investigation: the analytical model, which refers to the mathematical abstraction of the OpenDRIVE components, secondly the software design and development of the *XODR-Driver* and lastly, the use cases proposed for the evaluation and application of this work on the ITS context, particularly for *Traffic Management and Control*.

Chapter 1

Research Question and Gap

At the German Aerospace Center (DLR)-Institute of Transportation Systems, the geodata are vital component to operate different driving simulators and test vehicles so as to develop and evaluate driver assistant and automation systems. In this context the road description format ***OpenDRIVE®*** evolved as the de-facto standard for geometrical and logical representation of complex road networks. The increasing demand for generating OpenDRIVE from real-world scenarios requires an accurate and suitable integration of OpenDRIVE in *Geographic Information Systems* (GIS) applications. As well as the combination with traffic control elements on the sphere of *Intelligent Transportation Systems* (ITS).

Seeing that OpenDRIVE is a potential source of geographical data for different purposes, it was possible to oriented this work to some open issues on this topic. So a research problem is presented as the start point of this thesis for the *Master of Science in Transportation Systems* (*Verkehrstechnik*), of the *Technische Universität München* in collaboration with the *Deustches Zentrum für Luft- und Raumfahrt* e.V (DLR). The reminder of this work is organized as follows. Section 2 presents the State of Art. Section 3 introduces and describes the modeling and software design of the OpenDRIVE GDAL-Driver. Section 4 describes the results and discussion. Finally, Section 5 concludes this work.

1.1 Problem

The few available OpenDRIVE tools and editors are mostly commercial and offer insufficient support of common geodata or even none at all. In the GIS domain the modular open source library *Geospatial Data Abstraction Library* (GDAL/OGR) serves as a standard interface between heterogeneous raster and vector geo-formats. An extension of GDAL/OGR to support OpenDRIVE networks natively will close the gap between both the driving simulation and the GIS domain to offer completely new approaches of OpenDRIVE creation and processing to the driving simulation community.

The OpenDRIVE format describes the track-based road networks and the features along using an analytical formulation, this differs form the typical representation of geodata: vector or raster formats. This mathematical representation from OpenDRIVE hinders the manipulation, management and the storage of the road network data with conventional GIS methods. Furthermore, the standard GIS operations and further analyses, such as geostatistics, can not directly be performed on datasets or databases implementing this to the format and data representation.

In order to overcome these issues and gaps, it has been proposed under the supervision of the the DLR-Institute of Transpotation Systems **the extension of the Geospatial Data Abstraction Library (GDAL/OGR) for OpenDRIVE support in GIS Applications for Visualization and Data Accumulation for driving simulators.**

1.2 Objectives

1.2.1 Main Objective

To extend the *Geospatial Data Abstraction Library* GDAL/OGR by a driver offering read-support for OpenDRIVE files by converting its mathematical representation to Simple Features (vectorial representation) as defined by the *Open Geospatial Consortium* (OGC).

1.2.2 Sub-Goals

This objective has been divided into the following sub-goals:

- To generate the OpenDRIVE schema version 1.4 as an object representation from the mapped XML files.

- To model the mathematical representation for each element of the OpenDRIVE format as follows:
  - *Points,* for objects related to OpenDRIVE roads and roadside facilities, e.g. signals, controllers and traffic signs,
  - *Polygons,* for OpenDRIVE lanes, ramps and intersections, and,
  - *LineStrings and MultiCurves,* for the OpenDRIVE reference lines and lane boundaries (tracks).
- To integrate the OpenDRIVE *XODR-Driver* as a GDAL/OGR driver (functional prototype).

1.3 Research Question

As a result of the previews analysis, some key questions arise and its deep understanding can be helpful through the research process. At first sight, it is important to remark the expected outcomes for this master thesis: the mathematical model (geometries conversion), the software architecture (software model) and the code implementation of the OpenDRIVE driver and related geotools for complete data flow.

As a matter of fact, the mathematical model plays a fundamental role on the entire data process, since it describes the road network and the elements along as geometrical features on the vectorial plane. For instance, points, lines, polygons, multi-lines and so forth; therefore, some key research questions arise at this point:

- Which functions can be used for the definition of each OpenDRIVE element as *Simple Feature* on the vectorial plane?
- How the continuous function from OpenDRIVE can be discretized and moreover, how to define an accurate *sample rate* for this discretization process?
- After the validation of the models, how can be the XODR-Driver integrated to GDAL?

1.4 Methodology

The chosen methodology for conducting this investigation is based on *Waterfall model* [Larman and Basili, 2003]: the incremental development and design process for extensive projects. Following this approach, this investigation has being phased over three stages: from the conception of the problem to the implementation of the solution. The Figure 1.1 summarizes the requirements for the application design.

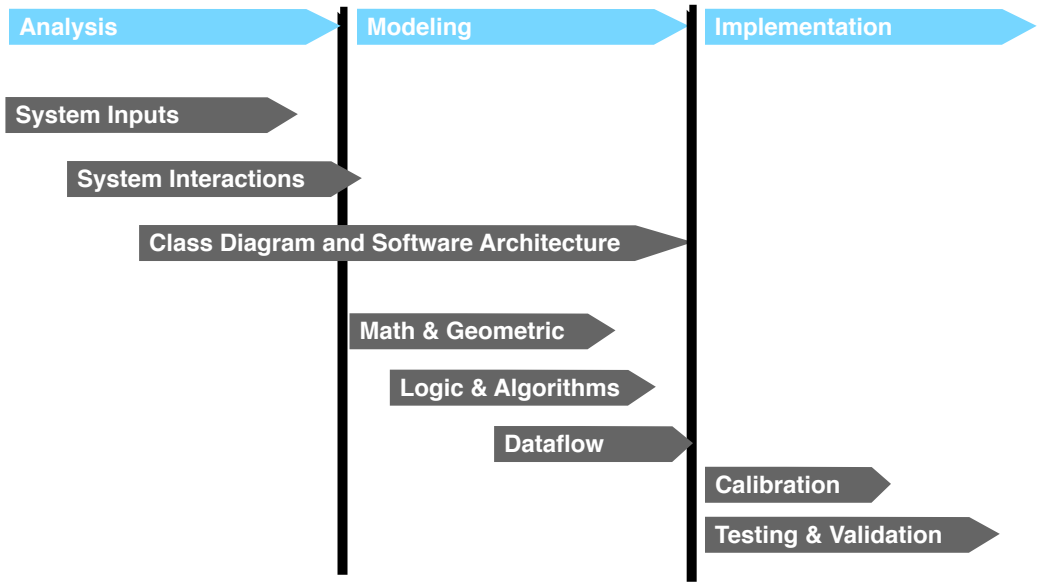


Figure 1.1: Phases of the research for this Master thesis.

i *Analysis and Definition phase*:

In this first phase, the system should be analyzed and understood, as well as the components, the characteristics and its interactions. The inputs and the desired outputs should be clearly characterized, moreover, the internal processes and states of the system identified. The *Black Box* model [Ljung, 1998] can help to have an overall idea of the data flow and the behavior of the logic system are at each state. This is shown by the Figure 1.2, which depicts the system dynamics from a general perspective.

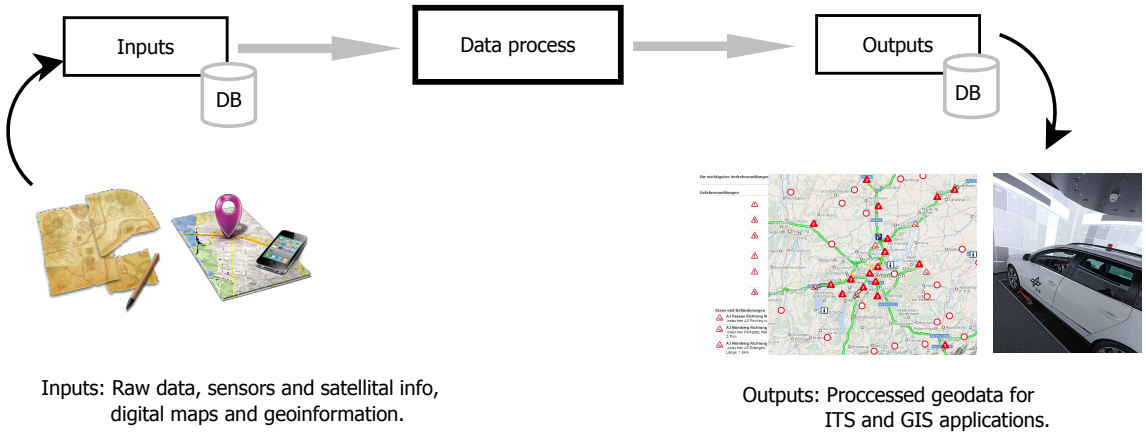


Figure 1.2: Black Box view of the data workflow for ITS and GIS applications.

ii *Modeling phase*:

In the modeling phase, the main tasks are to conceptualize and design the system on the basis of the previews stage. In this phase, the technical consideration are to be taken in account and precise definitions of particular states of the system are determined. The models are build following the specification, requirements and characterization of the definition phase and other observations. The purpose of this exhaustive technical analysis from the modeled system, is to prepare the components and the logic, for its implementation on a functional prototype.

In this case, for the mathematical model, different component must be considered, e.g. the set of variables, types of variables and functions, operations and so forth. On the other hand, for the the software models technical requirements must be stated. This can include: parameters, types and sources of data, decisions regarding the programming environment such as the programing languages, libraries, application programming interfaces, available source code for the development and so forth.

iii *Implementation and Testing phase*:

Finally, this last phase consists on the implementation and validation of the models as an software prototype. That is, the software design and development of the *OpenDRIVE driver*, from now called ***XODR-Driver***. In addition, testing and validation tasks are required in order to find possible error and to verify the outcomes of the system. For this sake, *use cases* will be employed; specially, for the verification of the functionality and performance of the *XODR-Driver* as part of GDAL and its applicability on the ITS field.

Chapter 2

State of the Art

This chapter introduces the conceptual framework of this investigation; from the basis of OpenDRIVE format to advanced techniques for geographical data manipulation and ***geo-informatics***. The first section offers to the lector an overview of the terms and definitions used on this work. The following section presents the Literature Review regarding the ***OpenDRIVE standard*** and lastly, the Related Work is presented in order to contextualize the lector with the latest development on this research field.

2.1 Definition of terms

This master thesis have been conducted considering topics on the areas of informatics, software design, mathematics, geographical information systems (GIS) and naturally transportation systems. Therefore, the need to establish a solid conceptual baseline is essential regarding the terms and thematics part of this research. Since each of these areas of knowledge is a science *per se*, this section focusses on the specific terminology used along this document. Nevertheless, the lector is invited to deeply explore the stated topics through the references.

- *Accuracy*: is one of the most important criteria of any GI system, which indicates the degree of closeness of a measured quantity to its actual value. In other words, the degree of veracity [Taylor and Cohen, 1998].

- *Class diagram*: in software, it is the representation of the dependencies and relationships of the elements in the Unified Modeling Language (UML) of a system.

This structure contains classes, attributes of these classes and its operations namely methods [Booch et al., 2008].

- *Data binding*: in software, it is the process of extracting the data from a direct representation of file and presenting it as as hierarchy of objects or events that correspond to a document vocabulary [CodeSynthesis, 2014].

- *Data model*: A data model is a set of constructs for describing and representing parts of the real world in a digital computer system [Longley et al., 2001].

- *Driver or GDAL-Driver*: in software, particularly in geoinformatics, it represents a translator for a specific format. In this context, the ***XODR-Driver*** enables the support of the OpenDRIVE *.xodr* files into the vectorial standard representation [Rouault, 2015].

- *Feature*: in the GIS context, a feature is a geograhic entity encoded using a given data model e.g. vector model. Features encapsulate the information regarding its characteristics $-attributes-$ and behavior $-functions/operations$. [Longley et al., 2001].

- *GDAL/OGR*: "GDAL stands for *Geospatial Data Abstraction Library*, is a translator library for raster and vector geospatial data formats that is released under an X/MIT style Open Source license by the *Open Source Geospatial Foundation*. As a library, it presents a single raster abstract data model and vector abstract data model to the calling application for all supported formats. Traditionally GDAL used to design the raster part of the library, and OGR -*OpenGIS Simple Features Reference Implementation*- the vector part for Simple Features" [Rouault, 2015].

- *Geographic Coordinate System (GCS)*: a GCS uses a three-dimensional spherical surface to define locations on the earth, typically with the *geodetic coordinates: latitude, longitude and elevation*. One of the common uses of the geographical coordinate systems is to geo-reference elements to locations (e.g. observed phenomena, geographical features and so forth) as well as to overlay data from diverse sources and perform GIS operations like fusion, matching among others. [Chang, 2006]

- *Object-Oriented Programming (OOP)*: in informatics, it refers to the software design and programming paradigm in which the data structure is based on the concept of *objects* and the associations among them. The idea behind this view, is to conceive

the data abstraction of the real world into a software data model and eventually into code; with the *attributes*, behaviors (*methods*) and *relationships* that represents the real system [Pierce, 2002].

- *Precision*: according to [Taylor and Cohen, 1998], precision is the degree to which further measurements or calculations show the same or similar results. In other words, repeatability or reproducibility of a measurement.

- *Polar Coordinate System*: A two-dimensional system where each point is represented as a distance r from the original and an angle θ. In the coordinate (r, θ), r is the radial coordinate and θ is the azimuth coordinate in radians. For more details, please refer to [Adams, 1991].

- *Rotation matrix*: in linear algebra, a rotation matrix is used to perform a rotation given an angle in the Euclidean space. During the rotation the original vector is rotated at an angle φ or rotation angle.

- *Translation*: in linear algebra, a translation is a parallel shift of the original vector. In the an Euclidean coordinate system, a translation is a function that moves the elements of a given vector a constant distance in a specified direction [Zimmermann, 2012].

- *Vector*: denoted by $\overrightarrow{v}$ is a quantity of more than one dimensions, that has *magnitude* and *direction*. The magnitude is the length and the direction is determined by the angle it makes with a horizontal line [Zimmermann, 2012].

- *Vector and iterator*: in informatics, a vector is a container or a storage location which represents arrays and can change the capacity dynamically (size). In contrast with the arrays, which are static. To access vectors are often used *iterators*, which are commonly pointers to the elements with properties and operators [Meyers, 2014].

2.2 Literature Review

2.2.1 Background

The OpenDRIVE project started when VIRES began building databases for driving simulators some years ago. These databases contained interfaces to the vehicle dynamics, to the simulated traffic flow and to the visualization elements conforming the road network and the system logic. In the year 2005, VIRES and Daimler worked together on the Daimler Driving Simulator in Berlin and proposed the standardization of the logical road description.

In this context, the need of a solid format and an standardized data container for the road elements for transport networks, led to the proposal of the OpenDRIVE standard revision on April of 2007. This new format aimed to facilitate sharing data between sources and types of simulator. After couple of years, other automobile manufacturers and research institutes joined to the OpenDRIVE consortium, at the moment are part of project the following members:

- *BMW Forschung und Technik GmbH*, Germany
- *Daimler AG*, Germany
- *Deutsches Zentrum für Luft- und Raumfahrt e.V.*, Germany
- *Krauss-Maffei Wegmann GmbH Co. KG*, Germany
- *Rheinmetall Defence Electronics GmbH*, Germany
- *TNO*, Netherlands
- *VIRES Simulationstechnologie GmbH*, Germany
- *VTI*, Sweden

Nowadays, the project is well known in the simulation, driving assistant and automated-driving community. Currently, it is considered a *de-facto standard* in the simulation industry. The Table 2.1 presents some of the companies and research institute using OpenDRIVE for commercial and scientific purposes.

Table 2.1: Some users and contributors of the OpenDRIVE project. Source: [VIRES, 2015].

2.2.2 OpenDRIVE: an overview of the Standard

OpenDRIVE® is the leading open format and de-facto standard for the description of the road networks in driving simulation applications.
Vires GmbH

OpenDRIVE is based on the *Extensible Markup Language* XML data structure. It defines the element and its hierarchy by means of tree-structured information. OpenDRIVE provides the detailed information of the road networks, organizing the data in categories such as roads, junctions, controllers and so forth. Then each element contains subcategories and associations between the elements e.g. a road segment contains traffic signs, and a traffic sign is managed by a controller. These data can be stored on databases and employed for different purposes, as shown on the Figure 2.1 some of the applications supported by OpenDRIVE are: online and offline traffic simulation, real-time evaluation of vehicle dynamics, real-time sensor simulation and driving simulation, among others [VIRES, 2015].

The typical data representation of a road network is given in terms of the *center lines* or ***reference lines***, which are commonly described by *vectors* or any other mathematical representation. In the case of OpenDRIVE, the roads and the elements among of them,

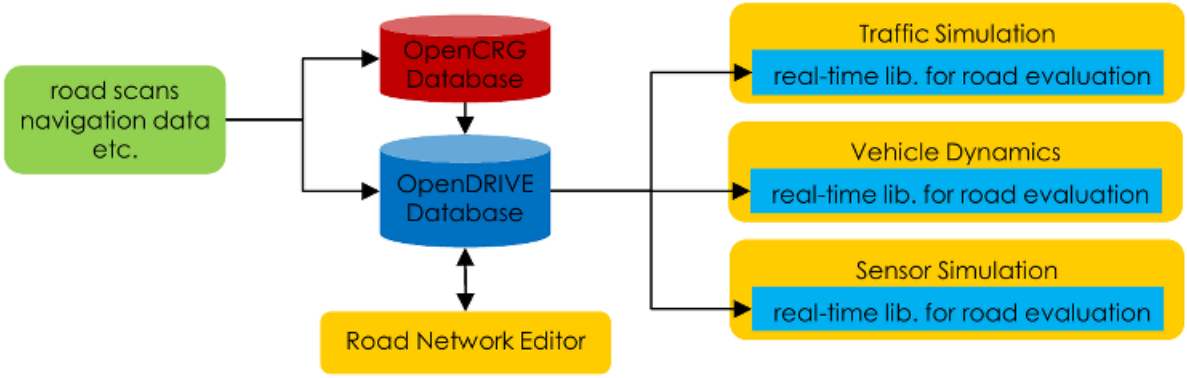


Figure 2.1: OpenDRIVE functional architecture. Source: [VIRES, 2015]

are defined by the arithmetical representation of *straight lines*, *arcs*, *spirals* and *polynomial* functions. The Figure 2.2 illustrates the disposal of the roads and the elements on the OpenDRIVE standard. Moreover, it is central to associate the OpenDRIVE components with the theory of *Geographical Information Systems*, which states that, geographic data link place, time and attributes [Longley et al., 2001].

In the OpenDRIVE context, the roads are defined by a reference line which defines the layout by means of the geometries and its geo-reference. The elements and properties along the roads such as lanes, traffic signs and elevation profiles, etc., are linked to a segment of the road as attributes or to the junctions i.e. intersections.

It is important to mention that, all the signalization elements along the road belong to a determined point linked to the reference line by the geographical coordinates. As an illustration the Figure 2.2 presents the diverse components of an OpenDRIVE road network, as mentioned before, the reference line containing the geo-location on the cartesian plane; the lanes and geometries that define the road layout and the elements along, which are traffic signals, traffic controllers, bus stops and other user-defined elements.

In addition to the elements provided by the standard, OpenDRIVE allows the extension of the the elements by the user-defined components. Because of the parametrization of the standard it is possible to associate or added new elements. This characteristic provides a level of extensibility to the standard to further applications and to the customization of the current version to certain types of simulation, studies on even vendor requisites.

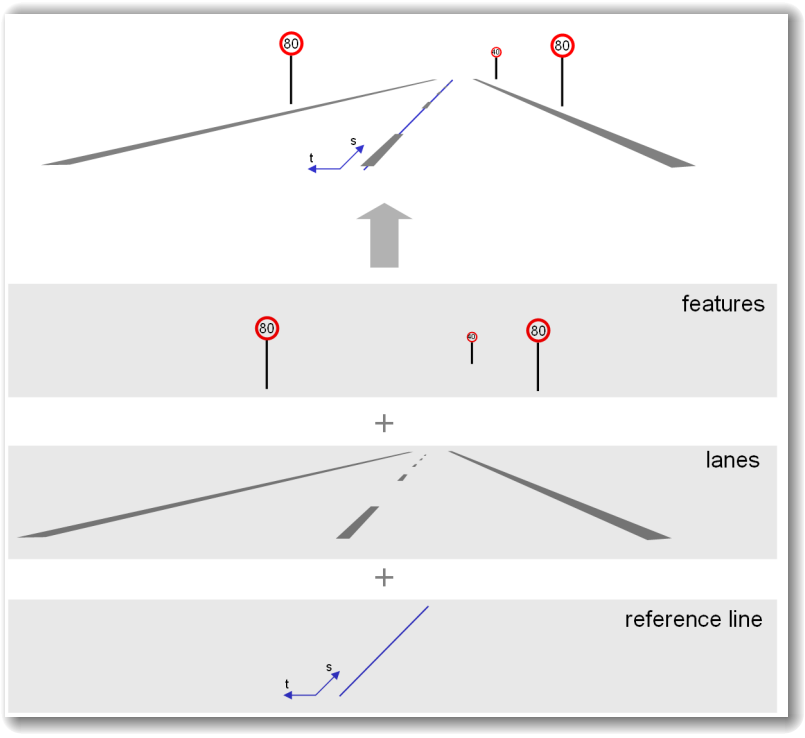
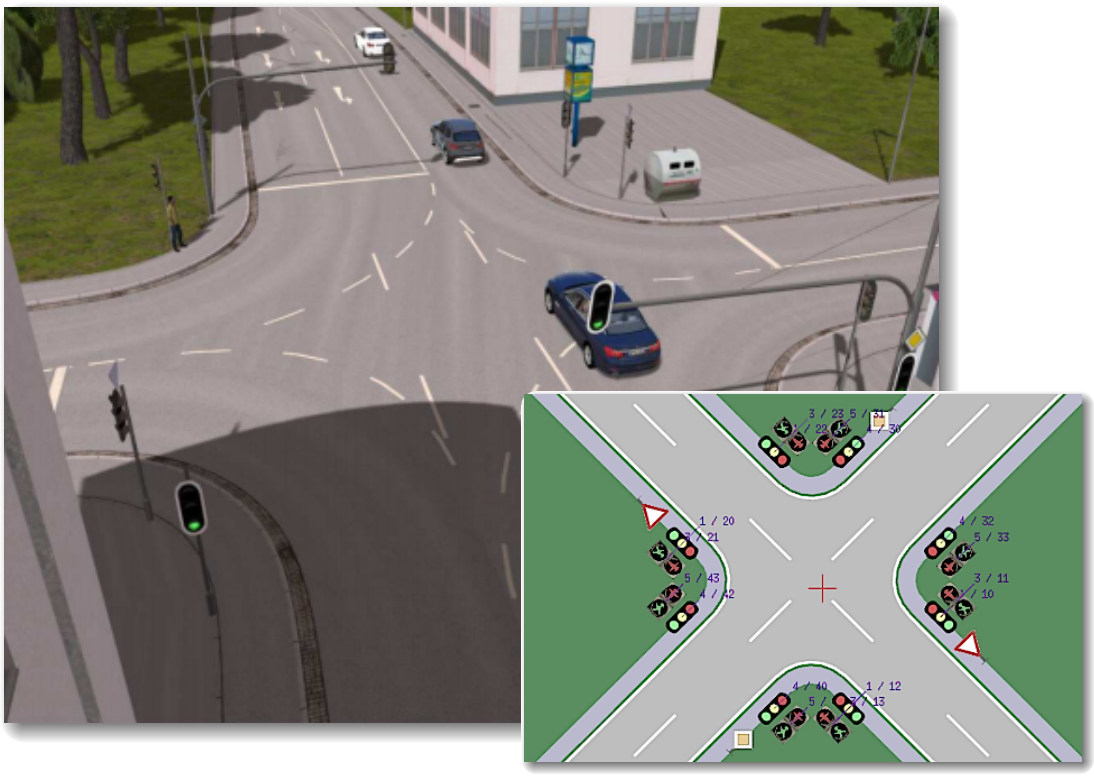


Figure 2.2: Components of the Road. Source: [VIRES, 2015].

2.2.3 OpenDRIVE for Traffic Management and Control

Considering the framework of this thesis, on the *Intelligent Transportation Systems*, new features are included on the latest version of the OpenDRIVE Standard Revision 1.4. These are especially useful for the definition of static and dynamic controllers, the signal programs, variable messages signs, as well as, other control components for traffic management and traffic simulation.

For this sake, according to OpenDRIVE Rev.1.4: "a ***controller*** provides the states for a signal group. A set of signals within a junction or a set of dynamic speed restrictions on a motorway. 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." [Dupuis and et. al., 2015]. As an illustration of the disposal of these control elements, the Figure 2.3b shows the controllers, traffic lights and signalization on a road intersection; in the same way the Figure 2.3a presents the 3D view of a typical intersection on the simulation models. In addition, the traffic signals may have Country Codes for the following countries: Austria, Brazil, China, France, Germany, Italy, Switzerland and USA.

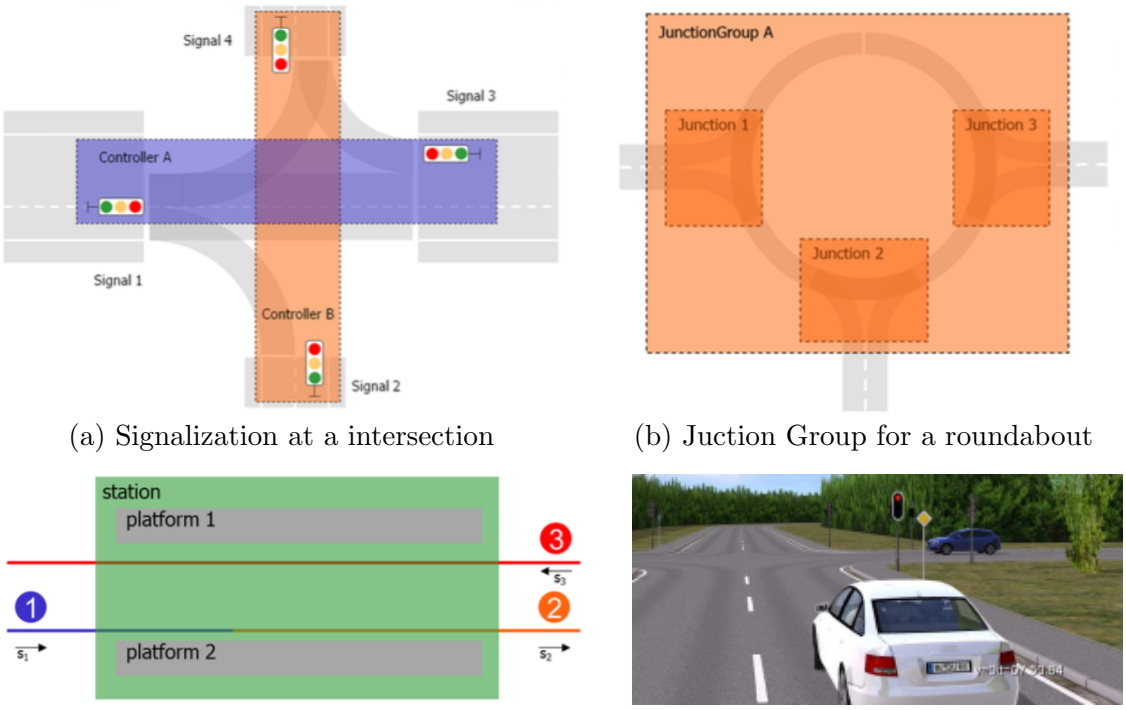


(a) Signalization at a intersection (b) Railway Stations

Figure 2.3: Traffic Control components of the part of the OpenDRIVE Standard Revision 1.4. Source: [VIRES, 2015].

For generic use, the code "opendrive" will be the default value. For individual signals, the signal type indicates the corresponding operational mode. For complete definition of the signal states provided by OpenDRIVE, please refer to the complete standard revision [Dupuis and et. al., 2015].

The way to group a set of traffic light is by means of a *junction*. A *junction group record* can be used to simulate a coordinated intersection (Figure 2.5a) and roundabouts (Figure 2.5b). The extensibility of the OpenDRIVE format in the field of Traffic Management and control is owned to the wide range of signalization and control elements available in addition to the *user-defined* components. The elements regarding public transport networks are also part of the OpenDRIVE standard such as train and bus stations, railways and tunnels (Figure 2.4c). Furthermore, the visualization of the traffic signs are illustrated on the Figure 2.4d within a 3D simulation scenario with OpenDRIVE.



(a) Signalization at a intersection

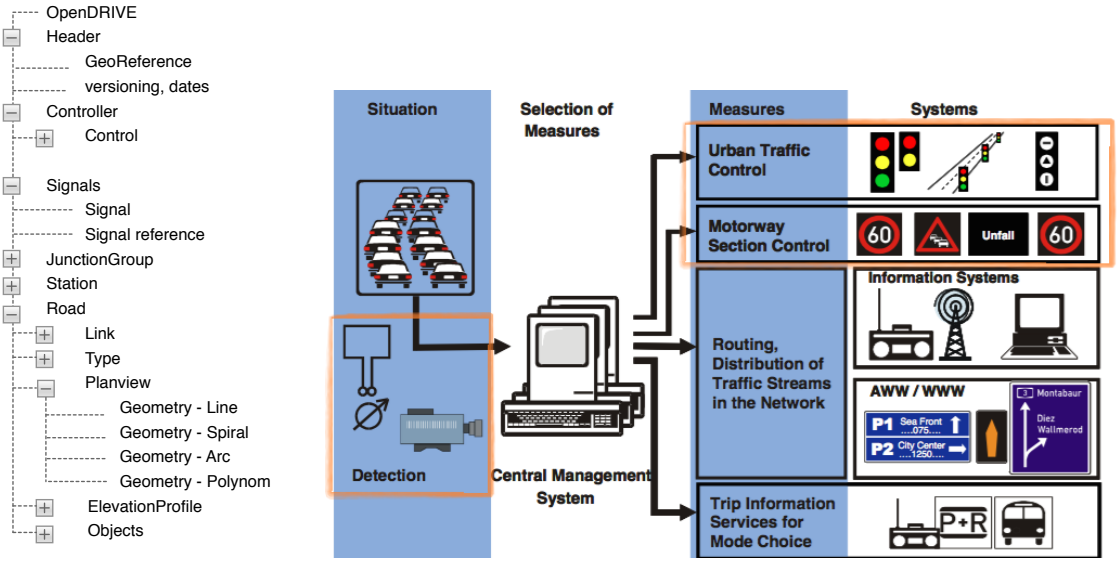
(b) Juction Group for a roundabout

(c) Railway Stations

(d) Simulation of traffic lights

Figure 2.4: Traffic control elements at OpenDRIVE. Source: [VIRES, 2015].

Considering the traffic control and management processes, the Figure 2.5 depicts the mechanism of the traffic control on all the levels: regulation and control, recommendation and guidance and information services. Linking the OpenDRIVE components with the ITS for traffic control mechanisms, it can be denoted that, for urban traffic and motorways section control, the standard provides the key elements for traffic control systems.



(a) OpenDRIVE (b) Traffic Management and Control systems.

Figure 2.5: Comparison of OpenDRIVE components and the Traffic Control Sphere

2.2.4 Related Work

In the recent years the use of the OpenDRIVE standard have notably increased, in consequence, the scientific research on this area follows the same tendency. In order to have a close examination of the previous work, this section presents some related investigations on the topics of modeling and visualizing the OpenDRIVE standard. This study will provide the baseline, namely the *State of the Art* concerning the geotools to manage and handle the OpenDRIVE data format.

Open-source road generation and editing software [Kurteanu and Kurteanu, 2010].

This work examines the domain of driving simulators and proposes an application, which can be used to generate logical and geometrical road data from the OpenDRIVE standard. The details of the standardized format used to store the logical road representation is described, as well as, the process and the problems encountered during the development of the application and its graphical user interface.

Automatic generation of OpenDRIVE roads from road measurements [Shi, 2011].

This project was part of the Swedish National Road and Transport Research Institute (VTI) in the area of Human Behaviour Analysis for transport systems. The goal was to integrate GPS data source and OpenStreetMaps [Haklay and Weber, 2008] with

the OpenDRIVE format. The software traces the roads described by OpenStreetMaps and converts the information to the OpenDRIVE data format. Citing the author: "The result was reasonable and good enough so that VTI could use this way and program to generate OpenDRIVE file that they want".

HORN-Hank and OpenDRIVE Road Networks [Öberg, 2012]

This thesis describes the Road Network Editor program HORN (HANK and OpenDRIVE Road Networks), developed for the HANKs scenarios (driving simulator of the University of Linköping). HORN is a software which aims to implement scenarios in a more efficient way, allowing to the user the ability to create larger scenarios. Before HORN HANK the scenarios were mostly modeled by hand and HORN tried to make the process more practical.

Design of a transport network for cognitive agents in virtual environments

Original title: "Konzeption einer Verkehrsnetzrepräsentation für kognitive Agenten in virtuellen Umgebungen"[Haubrich, 2013].

This work was carried out under the Agent-Based Traffic Simulation (AVeSi) Project, the traffic simulation for virtual environments. The target was to link the microscopic and mesoscopic views of traffic within a simulation approach. The transition between the two models required a transport (road) network which could be automatic generated. For this sake, OpenDRIVE was use as the data network format.

The overall objective of AVeSi is the development and implementation of a realistic traffic simulation for virtual environments through the use of psychological personality profiles. So not only a functioning traffic simulation has to be created as a basis, in which the participants adhere to the known traffic rules, but the behavior of the agents should also be extended so that these rules as in reality in certain situations and depending on the personality profile and the the driver's mood also transgressed [Seele et al., 2012].

OpenDRIVE Viewer The offered OpenDRIVE viewer is the official tool provided by Vires to visualize the *.xodr* files. Currently, works under Linux and do not allow the edition of the features of the road network. It is a simple tool to depict the XML files with restricted functionality. It is available to download on the OpenDRIVE download portal[VIRES, 2015].

Different from the previews work, we propose the design and development of the ***OpenDRIVE XODR-Driver*** providing a set of geotools for the data management and edition. The implementation included the translation model from the arithmetical native representation of the roads into the standard vectorial format of the GIS plane. As a result, the OpenDRIVE Driver will translate the road networks to the conventional GIS data container given by layers, datasets and the corresponding coordinate system. Thus, the *XODR-Driver* will enable functions such as opening, handling, visualizing and managing the files, on any GIS software platform like QGIS or ArcGIS. The contributions of this master thesis closes the gap, concerning the few geotools available for OpenDRIVE manipulation. Moreover, the integration of the *XODR-Driver* to the GDAL/OGR library extends the functionalities and operations of the OpenDRIVE datasets.

Chapter 3

Modeling and Implementation

The heart of any Geographical Information System is the data model,
which is a set of constructs for representing objects
and processes in the digital environment.
Paul Longley

This chapter presents the process of data modeling of the solution derived from this investigation. According to the employed methodology, three phases are distinguished: *analysis*, *modeling* and *implementation*, the Figure. 3.1 depicts the developing process of this work in gradual phases. In this scope, the process starts at the standard-based OpenDRIVE format and the understanding of the geographical and data model. The second step, consists on the definition and conceptualization of the data model of the OpenDRIVE road network and traffic control elements. The next stage comprises the software development and implementation of the models: the mathematical and logical abstraction of the geodata with an object-oriented design. The rest of this section will introduce the technical aspects of the processing and handling of the OpenDRIVE data for applications on the transportation sphere such as assisted driving, driving simulation, geographic databases as well as GIS purposes.

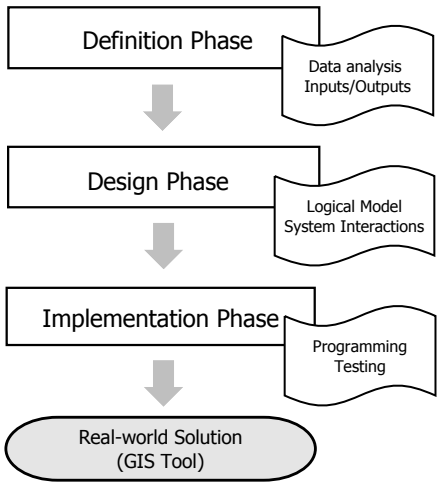


Figure 3.1: Phases of data modeling for the OpenDRIVE-Driver

3.1 Analysis: Data Model and Workflow

The process of data modeling specially on the area of *geographical information systems* requires a deliberately identification of the components of the system: inputs, outputs, processes and interactions. In view of the above, the physical model must be represent the real-world phenomena the closer as possible to the reality. And moreover, this model reproduce the phenomena in a logical and implementable system. For the context of GIS and considering its role in the *Intelligent Transportation Systems*, the fundamental problems with the spatial data are: what to represent the and how to characterize the geodata. In this investigation, these and some other related problems were faced.

On the scope of driving simulation and assisted driving, and also other application such as dynamic routing for individual of guided routing, traffic management and control, the geodata play a fundamental role. As inputs for many real-time applications, the systems demand accuracy, reliability and precision of the geographic information.

On the automotive field, *OpenDRIVE* is considered the standard de-facto for driving simulation on the assisted driving testing; *OpenDRIVE* allows the data handling of transport networks elements such as an inventory of the elements, its analysis, mapping and geo-referencing. In this context, the source the spatial information are *OpenDRIVE* files, which contain the road network information such as reference-roads lines, lanes, controllers and so forth. The source data is represented as *Extensible Markup Language* (XML) and the extension of the files are defined by Vires as ***.xodr***.

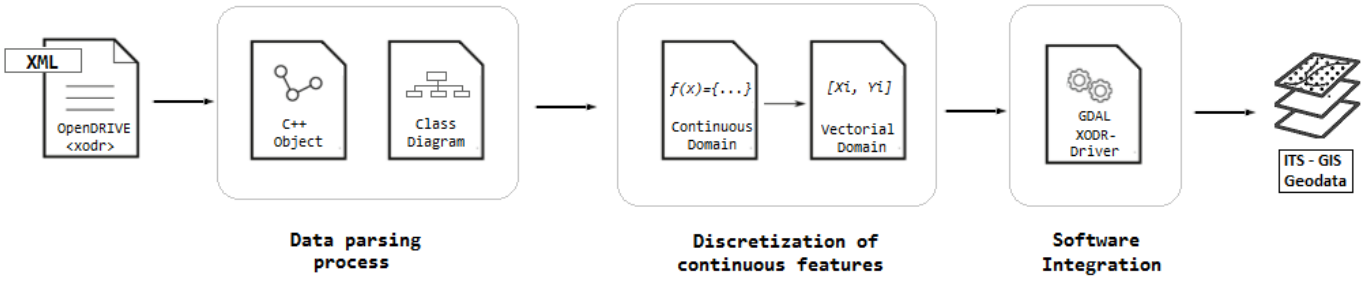


Figure 3.2: Data workflow for OpenDRIVE geodata

Hence, the standardization of this data format can lead to an effective and accurate manipulation of the geodata, in addition, the aggregation with other sources of information like cadastral data, Origen-Destination (OD) matrices, spatial distribution of particular phenomena and so forth. So that, the aim of this work is to develop a *driver* or format converser from OpenDRIVE to the standard GIS vectorial representation, namely **XODR-Driver**. The data workflow of this investigation is depicted on Figure 3.2 and along this chapter, where it will show how each phase and process were undertaken.

To start, an abstraction of the elements of OpenDRIVE is required, for this sake, a suitable representation of the data on the vectorial domain should be modeled. In other words, mapping from roads, tracks, lanes, control components, etc. to its vectorial form recording the characteristics and its meaning. The coding of each road element to points, polylines and polygons requires a detailed understanding and identification of these components as ***features***. All the roads are mathematically described by a *reference line*, which is defined by a geometry representation as a continuous function. A long this reference, all the elements are disposed: elevation profiles, lanes, junction areas, traffic signs, control components and so forth.

Answering to the question of how to represent each road element in the standard GIS data structure, the components were identified, from the OpenDRIVE schema with its attributes. The Table 3.1 shows the variables used by OpendDRIVE and the corresponding unit from the International System (SI) considered on this work. For the sake of this investigation, only the elements regarding the track of the road and the controllers will be considered. The OpenDRIVE format offers the description for the fields and elements that are characterized the road and its elements.

Table 3.1: Naming Convention and Units of the OpenDRIVE Standard

Category	Description	Unit
Distance	Meter	m
	Kilometer	km
Speed	Meter per second	m/s
	Kilometer per hour	km/h
Acceleration	Meter per second squared	m/s^2
Angle	Radians	rad
Geo-Referencing	Projection in Well-Known Text format	*WGS84*

3.2 Data Structure and Logical Representation

The mapping from the pure XML data format to the GIS standard representation, implies the modeling of the element of the OpenDRIVE domain. For this purpose, a parsing process of the original XML schema to an object-oriented approach was performed. This process is called ***data binding*** [Mutschler III and Stefaniak, 1999]. It consists on the generation of a *class diagram* and instances of these *classes* as *objects*. This data structure represents the vocabulary contained on the XML files, in this case, for the *C++ programming language*.

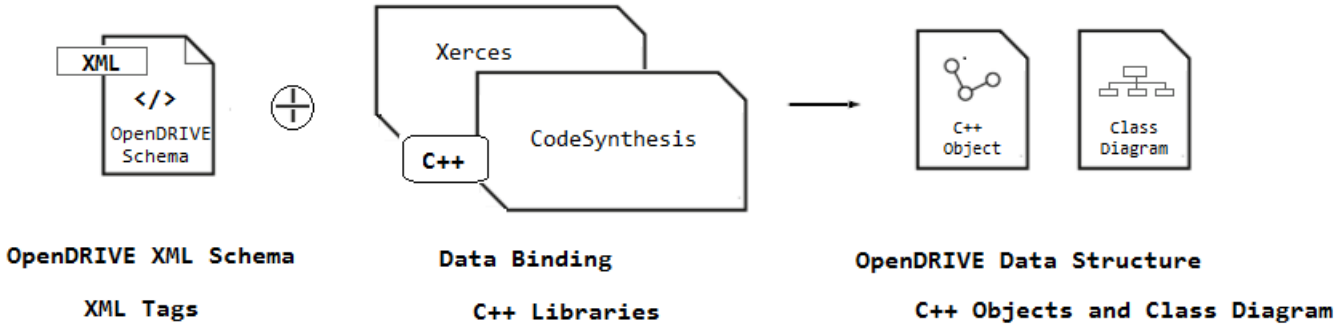


Figure 3.3: Data Binding Process

The advantages of creating an in-memory allocation data structure, with the hierarchy of objects instead of reading theXML files for each software routine are: in first place, the generation of the class diagram, with the relationships between classes and object. Secondly, the disposal of the related relational model for databases e.g. *PostGIS* [Obe and Hsu, 2011].

Moreover, the data are allocated in-memory, providing access during the runtime. The data binding provides high performance, modularity, and scalability to the *XODR-driver*. In terms of the data handling, it is fundamental to ensure the the integrity of the data, for this sake the object-oriented data structure provides the capabilities for maintaining the data tree-structure of OpenDRIVE.

For this master thesis and the development of the *XODR-driver*, two open source libraries have been used for the data binding process: *CodeSynthesis* and *Xerces*.

CodeSynthesis [CodeSynthesis, 2014]

> "It is an open-source, cross-platform XML schema to C++ data binding compiler. Provided with an XML instance specification, it generates C++ classes that represent the given vocabulary as well as XML parsing and serialization code."

Xerces [Xerces, 2015]

> Xerces-C++ is a validating XML parser written in a portable subset of C++ by the Apache Software Fundation. This library provides the ability to read and write XML data. Moreover, this robust shared library provides C++ and Java methods for parsing, generating, manipulating, and validating XML documents.

As a result of the data binding process of the OpenDRIVE schema, the comprehensive class diagram was generated (See Section 3.4). With the classes and data structure, the capability of read the information from the *".xodr"* files is enable. Subsecuently, the next step is to model the geodata on the geometrical plane, in other words, to build the mathematical abstraction of the geo-referenced phenomena i.e. road elements, contained on the OpenDRIVE standard.

3.3 Modeling OpenDRIVE: An analytical approach

This section focuses on the mathematical model of the OpenDRIVE reference lines. First, the classification of spatial phenomena i.e. roads and the elements along into their mathematical representation. Second, the discretization and sampling to the vectorial plane and finally, the construction of the GIS model from OpenDRIVE.

From the theory of GIS, is known that there are four types of model for representing objects on the geometrical plane: *vector*, *raster*, *graph* and *hybrid* models. The election of the type of model depends on several criteria, e.g. the properties of the set of data, the functionalities, the structure of the geodata and its applications. Following this selection method, the election of a particular style of the data model i.e. *vector model*, relays mostly on the use and applicability of OpenDRIVE. In this case, the simulation purposes for driving and traffic models, assisted driving and data fusion and data completion point to the vectorial representation of the OpenDRIVE standard.

3.3.1 The Geometry of OpenDRIVE

As it was mentioned before, OpenDRIVE describes the geometry and disposition of the roads and the elements that are part of it. For this aim, the essential component of OpenDRIVE is the ***Reference Line (Track)***. The track provides the analytical formulation of the geometry of roads as well as features along the roads (e.g. lanes, signs, signals).

On a global scope, the reference line can provide all the information to create the road network, including the geographical coordinates and the location of the elements in the track. The Figure 3.4 is a satellite image of a roundabout, it depicts an sketch of the reference lines as an illustration of the track configuration. It is possible to observe the geometry and the geometrical structure of the track: straight lines, curves, circles and arcs (semicircles) provides the mathematical formulation of the reference line. Another observation from this areal photo, shows that a set of functions (geometrical figures) shape the roads, lanes, intersection and so forth.

In the case of OpenDRIVE, the geometries are given by continuos functions that define the layout go the reference line in the ***x/y-Plane*** or *Plan view*. Consequently, four geometric elements record the position arrangement of the track: *straight lines*, *arcs*, *spirals*, and *cubic polynomials*.

Figure 3.4: Foto Satellital

- *Straight lines*: this element is given by the *start position* for the dimension x and y and the *length*.

- *Arcs*: a constant *curvature* $[1/m]$ describes a semicircle of a given *length*. The arcs circumscribes the curves on a road segment. When the curvature is positive, then it forms a *left turn*; for the case of a negative curvature, a *right turn* is described.

- *Spirals*: the parameters used to compute the clothoids are the *initial curvature* and *end curvature* for the *Euler Spiral* [Adams, 1991] or other if specified. Spirals in OpenDRIVE are used as transition curves on the road.

- *Cubic polynomials*: The polynomial is calculated in the local coordinate system given by the Equation 3.1, where a, b, c, and d are coefficients.

$$v_{local}(du) = a + b * du + c * du^2 + d * du^3 \tag{3.1}$$

Figure 3.5 depicts the geometric elements to describe the reference line of the roads on the OpenDRIVE domain. It can be observed that the reference line is shaped by a sort of geometries of different kind. The roads are identified with the reference line segment, which provides the geographical coordinates by continuos functions. The concept of curvature provides the direction of the turning points (curves), it will be deeply explained further on in this document.

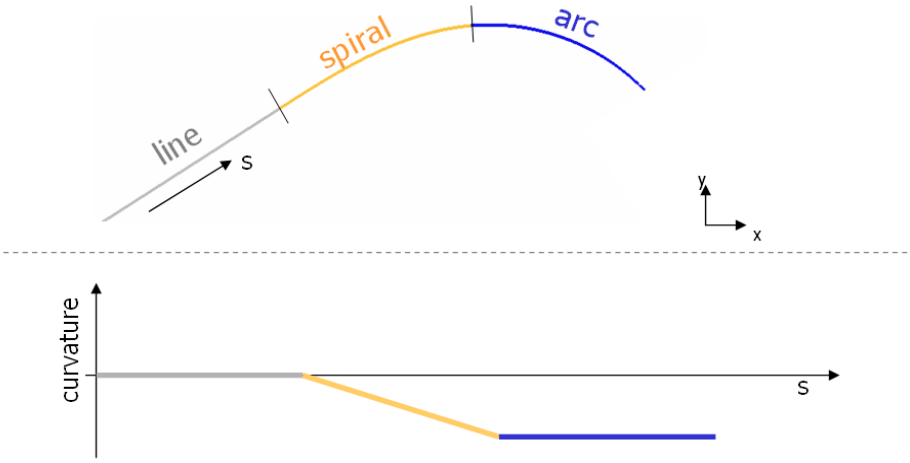


Figure 3.5: Geometry of the OpenDRIVE reference line.

The following subsection explains the discretization and sampling processes and methods, employed for the conversation of these continuos functions to the R^2 vectorial plane.

3.3.2 OpenDRIVE on the Vectorial Plane

The geographic representation of data enables tasks like mapping, visualization, querying databases, spatial pattern identification and performing analysis over the datasets. So that, the geodata models must provide the suitable types of data and object representation to undertake this functions and applications. In this context, OpenDRIVE provide a set of discrete objects such as roads, signs, signals, etc; therefore the more suitable alternative

to represent OpenDRIVE road networks is the discrete conceptual perspective i.e. *Vector Model*.

The road layout from OpenDRIVE, as previously discussed, comprised by the set of geometries of the *reference line* and provides an analytical formulation on the mathematical continuous domain. So ***discretization*** of the geometries through ***sampling processes*** are required. In addition, due to the nature of OpenDRIVE the ***geo-referencing*** of the data must be conducted on the Two-dimensional plane (Coordinates x, y) namely R^2 Plane.

For this thesis, we have modeled the mathematical and geometrical baseline for the discretization of straight lines and arcs from the reference lines, this was conducted as follows:

Lines to vector

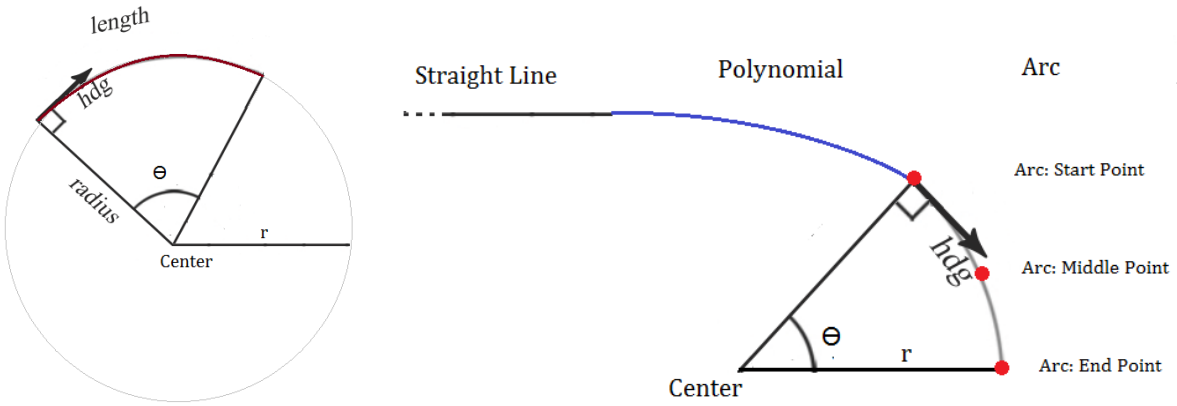
For the straight road segments, two points are required: the start and the end point. The *start point* is given on the geometry description as (x_i, y_i). On the other hand, the *end point* (x_e, y_e) is calculated from the given polar coordinates with the value of the *heading* (hdg) in radians and the *length* according to the Equation 3.2:

$$Coordinates(x_e, y_e) = \begin{bmatrix} x_i + \cos(hdg) \times length \\ \\ y_i + \sin(hdg) \times length \end{bmatrix} \tag{3.2}$$

Given the starting and the ending coordinates of the straight line, a linear interpolation between these points defines the points in between.

Arcs to vector

The arcs are semi-circles that circumscribes the turns of the road with a given constant *curvature* C. The start point (x_i, y_i) and *heading* (hdg) in radians are also known values of the geometry. In order to sample this segment of circle, three main points are to be calculated: the *initial point* (x_i, y_i), *middle point* (x_m, y_m) and *end point* (x_e, y_e). The discretization algorithm will select the points along the arc according with a given ***sample rate***. The OpenDRIVE standard provides the initial coordinates on the X-Y plane and the *heading* as polar coordinates with center of circle (x_0, y_0). Thus, some Euclidean geometry concepts and linear algebraic operation over the vectors are required so as to find the middle and the end point. The arc



(a) Arc Geometry (b) Road geometries

Figure 3.6: Geometrical representation of OpenDRIVE

belongs to the circle with *radius* r and $circumference$ given by the Equation 3.3 and depicted on the Figure 3.6.

$$Circumference = 2 * \pi * r \tag{3.3}$$

The *radius* r can be derived from the *curvature* C from the Equation 3.4:

$$|C| = \frac{1}{Radius} \tag{3.4}$$

The angle θ of the arc is calculated with the segment *length* L and the *radius* r as follows in the Equation 3.5:

$$\theta = \frac{Length}{Radius} \tag{3.5}$$

Based on these equations and considering the value of the curvature (positive or negative), the coordinates for the start, middle point and end points are calculated within the ***Polar Coordinate System*** by the following equations:

For positive curvature ($C > 0$):

$$\textit{Start Point } (x_i, y_i) = \begin{bmatrix} r \times \sin\theta \\ \\ -r \times \cos\theta \end{bmatrix}, \text{ with } \theta = 0 \tag{3.6}$$

$$\textit{Middle Point } (x_m, y_m) = \begin{bmatrix} r \times \sin\dfrac{\theta}{2} \\ \\ -r \times \cos\dfrac{\theta}{2} \end{bmatrix}, \text{ with } \theta = \frac{L}{R} \tag{3.7}$$

$$\textit{End Point } (x_e, y_e) = \begin{bmatrix} r \times \sin\theta \\ \\ -r \times \cos\theta \end{bmatrix}, \text{ with } \theta = \frac{L}{R} \tag{3.8}$$

Owing to the design of the the turns of OpenDRIVE, the sign of the *curvature* indicates where to start the drawing of the chord: if positive, from $-\pi/2$ and for negative from $\pi/2$. The equations for negative C ($C < 0$) are given as follows:

$$\textit{Start Point } (x_i, y_i) = \begin{bmatrix} r \times \sin\theta \\ \\ r \times \cos\theta \end{bmatrix}, \text{ with } \theta = 0 \tag{3.9}$$

$$\textit{Middle Point } (x_m, y_m) = \begin{bmatrix} r \times \sin\dfrac{\theta}{2} \\ \\ r \times \cos\dfrac{\theta}{2} \end{bmatrix}, \text{ with } \theta = \frac{L}{R} \tag{3.10}$$

$$\textit{End Point } (x_e, y_e) = \begin{bmatrix} r \times \sin\theta \\ \\ r \times \cos\theta \end{bmatrix}, \text{ with } \theta = \frac{L}{R} \tag{3.11}$$

It is important to remember that, at this point the vectors belong to the polar coordinate plane and three operation must be performed in order to obtain the coordinates on the ***Cartesian Coordinate System***. The operations are part of the vectorial geometry and are explained as part of the OpenDRIVE discretization process: *translation*, *rotation* and *translation* in the vectorial plane to the OpenDRIVE domain.

In the first place, the Translation vector $\vec{T}$ operates the *Start Point*, *Middle Point* and *End Point*. These vector points will be notated from now on as $\vec{P}$ (from Points), as a result the *Translated Vector* $\vec{P^T}$ is obtained by the Equation 3.12:

$$\mathbf{P^T} = \mathbf{T} + \mathbf{P} \tag{3.12}$$

$$\text{with } \mathbf{T} = \begin{bmatrix} 0 \\ r \end{bmatrix} \text{ for } C > 0 \text{ and with } \mathbf{T} = \begin{bmatrix} 0 \\ -r \end{bmatrix} \text{ for } C < 0.$$

Secondly, we need to rotate the vectors $\vec{P}$ in function of the orientation φ, in this case given by the *heading* in radians. For the two dimensional plane, the *Rotated Vector* $\vec{P^R}$ is given by the product (matrix multiplication) of the the Rotation vector $\vec{R}$ and the Translated vector $\vec{P^T}$ by the Equation 3.13:

$$\mathbf{P^R} = \mathbf{R} \cdot \mathbf{P^T}, \text{ with } \mathbf{R} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \tag{3.13}$$

Finally, one last translation operation is required. The translation from *Polar Coordinates* with center on the origen $\vec{O^C}$ $(0, 0)$ to the OpenDRIVE coordinate system, namely Cartesian Points vectors $\vec{P^{TT}}$. The *Translation Vector* $\vec{T}$ is given by the Equation 3.12 and the points for the arc on the cartesian coordinates are given by the following Equation 3.14:

$$\mathbf{P^{TT}} = \mathbf{P^T} + \mathbf{T} \tag{3.14}$$

As mentioned, the mathematical and geometrical models of the OpenDRIVE layout lead to have a set of scattered points (geographical coordinates) which portraits the elements of a road network. Nevertheless, we do not have any object presentation from the geodata so far. Seeing that, the next phase consists on the creation of 'content-meaningful' objects better known as ***features*** in the field of Geographic Information Systems.

3.3.3 OpenDRIVE to Simple Features

This section explains the construction of the digital model for the OpenDRIVE standard, in order to code the geographic information - vectors or coordinates (x, y) - into ***features***. The term *features* is linked to discrete objects namely entities, with characteristics or *attributes* on the conceptual vector model. From this perspective, we can coin the concept of feature to the object of this study: the roads belonging to OpenDRIVE; thus, in this context, a ***road*** will be connoted as a *feature*.

The datasets encoded under the vector approach are classified in: *points* (a pair of X and Y coordinates). When these points are connected by straight lines, comprise *polylines*, which can describe curves. And *polygons*, for the representation of areas. So, the features are vector objects which can be represented by these types of geometries namely ***Simple Features***. In consequence, this representation method is considered an efficient technique to capture and portray an *accurate* abstraction of the real world. The *XODR-Driver* employs the geometry classes and the architecture provided by the ***Open Geospatial Consortium (OGC)*** to represent the OpenDRIVE components. It is depicted on the Figure 3.7.

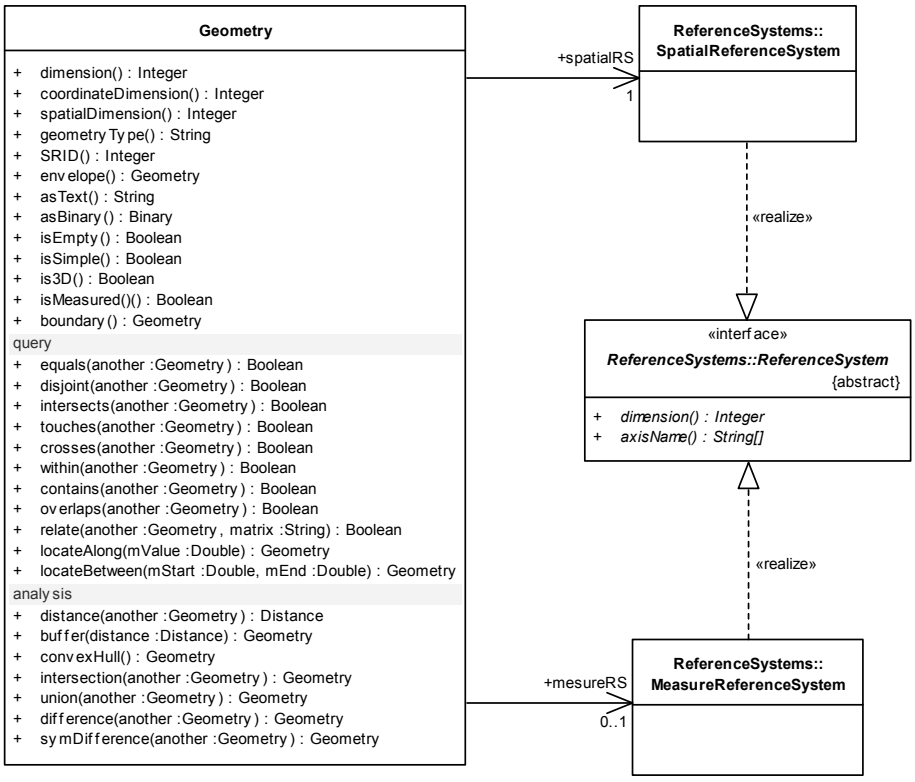


Figure 3.7: Geometry class description. Source: Open Geospatial Consortium

As discussed on the previews sections, the reference line can be distinguished by the geometrical shapes given as straight lines, curves and transition curves. At this points, we count with a set of points recording the geographical location under a certain *Geographical Coordinate System*. The current phase consists to create *Simple Features* instances from each discretized road given its spatial coordinates (x_k, y_k). The process of discretization and sampling of the road sections according to their geometry types is described by the Algorithm 1. The procedure starts with the reading of the *.xord* files and the data binding of the road networks as in-memory objects (See Section 3.1).

Algorithm 1 Creation of Simple Features from OpenDRIVE

```
procedure LOAD: OPENDRIVE ROAD NETWORK
    for each Road_k do
        Look for the Collection of Geometries
        for each Geometry_n do
            Look for the Class and cast
            if Geometry_n equals Line then
                Create OGRLineString with:
                StartPoint and EndPoint [(x_i, y_i),(x_e, y_e)]
            end if
            if Geometry_n equals Arc then
                Create OGRLineString with:
                StartPoint, MiddlePoint, EndPoint [(x_i, y_i), (x_m, y_m), (x_e, y_e)]
                SampleRate (in grads), Type of discretization.
                with the function: OGRGeometryFactory::curveToLineString(args[ ])
            end if
            STORE: OpenDRIVE Geometry_n → OGRLineString
        end for
        STORE: OpenDRIVE Road_k → std::vector<OGRMultiLineString>
    end for
end procedure
```

Due to the sampling process, the reduction of the data volume is prominent in comparison to the *raster model*. Moreover, the optimization of the in-memory access lead to faster computational time despite the data volume and complexity. Afterwards, with the $C++$ data structure and data loaded, it is possible to iterate a collection of *Roads*, this task is performed using a powerful programing tool called *vector<iterator>*. For each $Road_k$ element, a *cast* method is performed. In oder words, to identify and convert to the original *class* type based on the corresponding *geometry type*.

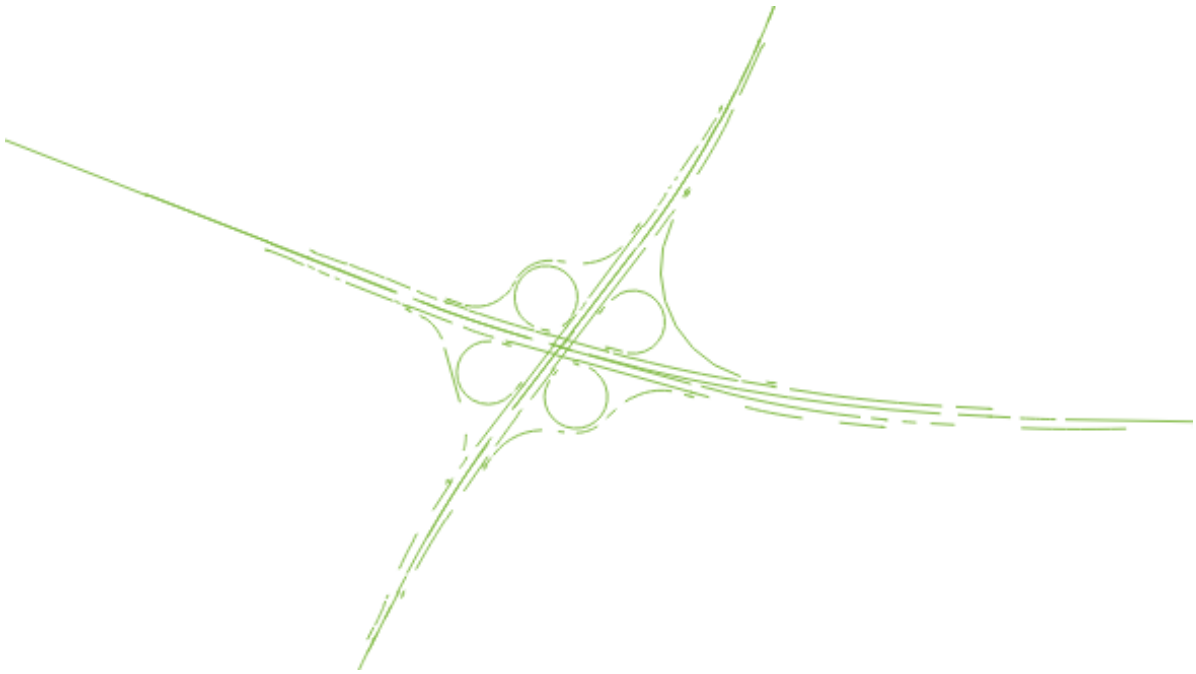


Figure 3.8: Intersection of the City of Braunschweig represented as Simple Features.

After the verification of the type of geometry for this section of road, the parameters of the *reference lines* are retrieved from the memory with a *pointer* from the address in-memory of the system. With the parameters and the type of geometry, the points are calculated defining the geometry for each segment of the track. To end up, with a collection of *OGRLineString* contained on a *OGRMultiLineString*.

The application of the described optimization techniques contribute to the performance of the *XODR-Driver*. Particularly, in terms of the computing time and the memory allocation required for the data storage. These are critical requirements for an effective system operation, taking in mind, that in general, geodata consist in a remarkable amount of information. As an illustration of the capabilities of the developed *XODR-Driver*, the Figure 3.8 shows an intersection of the City of Braunschweig represented as Simple Features employing the logic of the Algorithm 1 (the original *.xodr* file was provided by the *German Aerospace Center*).

3.4 Software Development of the OpenDRIVE Driver

After the modeling of the geometrical representation of the roads as *Simple Features*, it is now convenient to integrate the logical functionality into the ***Geospatial Data Abstraction***

Library (GDAL). So, providing advanced techniques for the spatial analysis and GIS data management. This section undertakes the technical approach concerning the software development and implementation of the proposed theoretical model for the data workflow, called the ***XODR-Driver*** (See Figure 3.2).

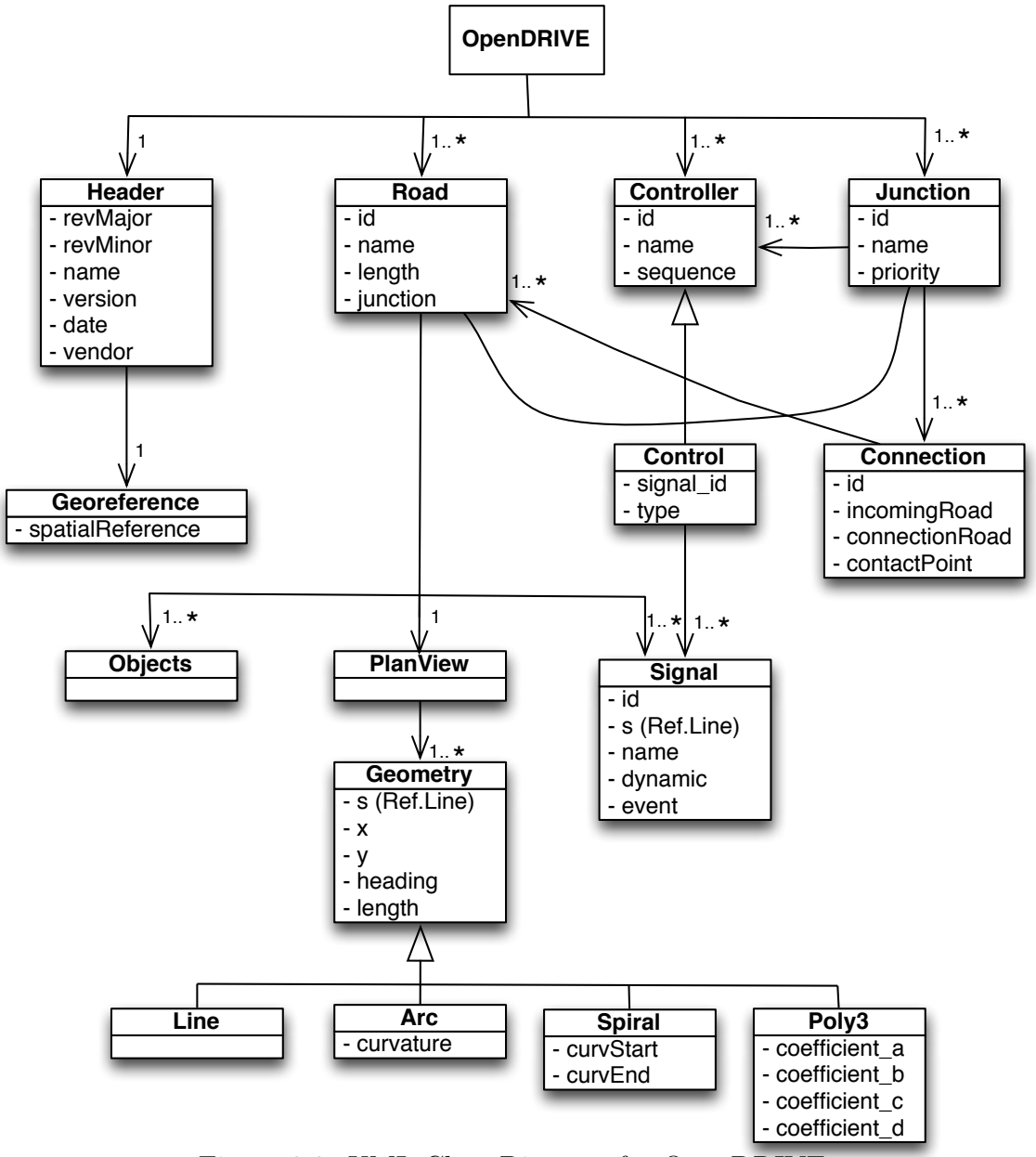


Figure 3.9: UML Class Diagram for OpenDRIVE

The software architecture of the *XODR-Driver* is based on the *Class Diagram*, in which is possible to observe from an *object-oriented* perspective, the logical structure of OpenDRIVE. This includes the whole set of road elements, and what it is more important,

the relationships and behavior that shapes the OpenDRIVE context. The *Class Diagram* is presented as part of the of software design phase and due to its complexity, only the relevant elements for this thesis are shown on the Figure 3.9.

These objects and its *attributes* and functionalities i.e. *methods*, provide the facilities to access, compute and modify the values of the GIS entities such as the road parameters and the controllers information. The objective is to extended the GDAL-OGR library (which can read-write many of the standard and well-known raster and vector formats of geodata) by means of the software implementation of a new module, namely *driver*, which adds the XODR format to the *OpenGIS Simple Features Reference Implementation* GDAL/OGR library as part of the core source code.

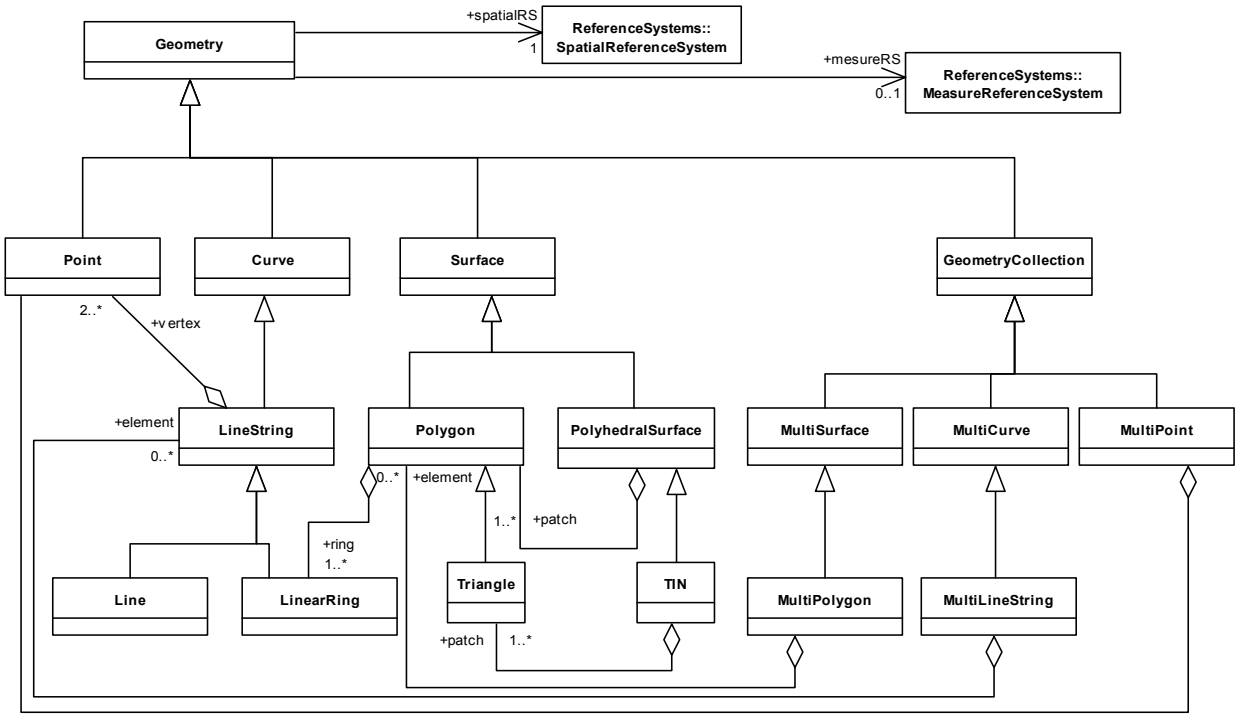


Figure 3.10: Simple Features class hierarchy Source: Open Geospatial Consortium

To start with the election of the programming languages. Given that GDAL/OGR is written in *C++*, it was adopted as the programming language for the source code of the *OpenDRIVE XODR-Driver*. Following this decision, the election of the programming framework was also decided based of the applications of the OpenDRIVE driver. In this case, the command-line tools and the vectorial operations on a Windows environment for further uses led to select *Visual Studio C++* as the programming framework. Nevertheless, it is fundamental to mention, that since we developed the source code of the *XODR-Driver*,

the compilation setup and the program configuration, it is totally platform independent, giving portability and robustness of the XODR source code. In short, the *XODR-Driver* can be compiled, linked and build under any platform or operating system.

The system architecture of the *XODR-Driver* is shown on the Figure 3.11, where the the level of integration of XODR to GDAL/OGR can be observed. The developed XODR modules were coupled to the core of GDAL, allowing the access to all native functionalities of the geospatial library, which includes the processing engine and the techniques for data analysis and handling methods.

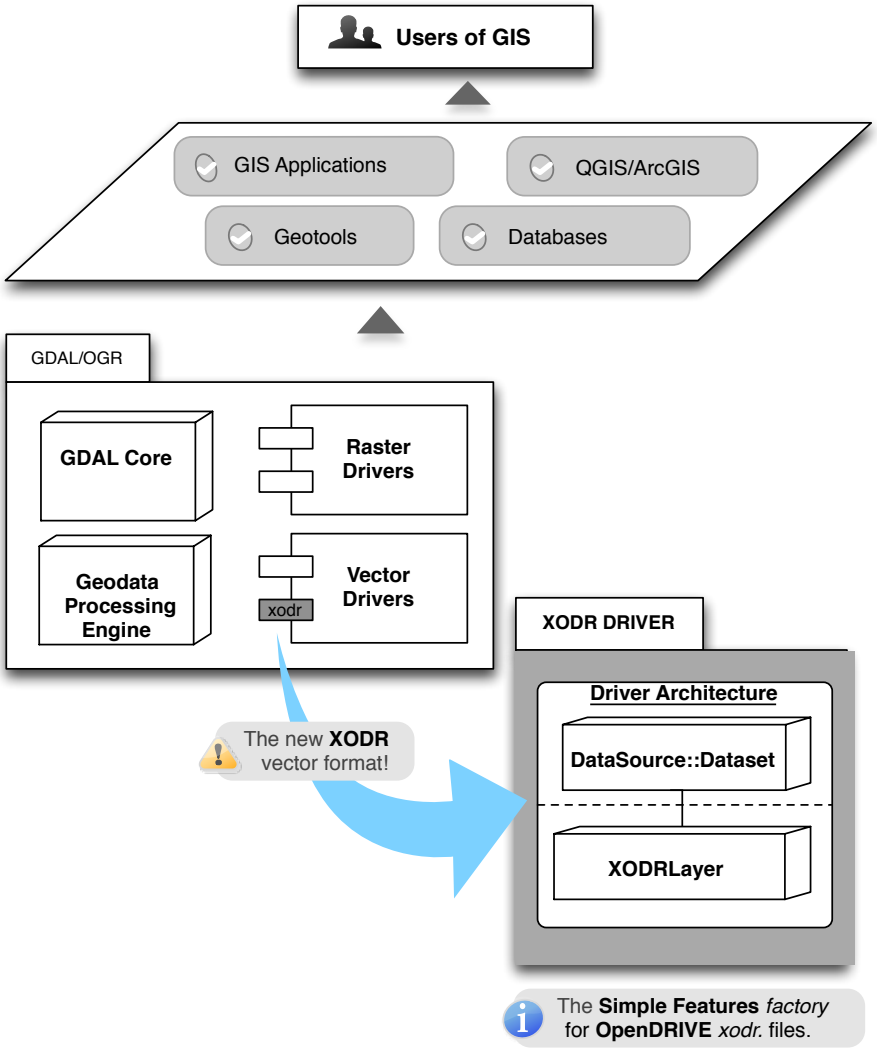


Figure 3.11: *OpenDRIVE XODR-Driver* Architecture

The following chapter presents the the results of this work from an implementation-oriented perspective: on the areas of *ITS* and *Traffic Management and Control*.

Chapter 4

Results and Discussion

In the previews chapters we identified, represented, and distinguished some of the OpenDRIVE elements as Simple Features on the vectorial GIS standard format with its corresponding georeference system; up to this point, following a theoretical and scientific method. Now in this chapter, we test and conduct experimental applications using our developed *XODR-Driver* on the *Intelligent Transportation Systems* field from a practical approach. For this sake, the *Use Cases* methodology was employed. Two cases are presented: in first place, the conversion of OpenDRIVE to ESRI Shapefiles or other formats; and secondly, the matching of traffic data with OpenDRIVE for Traffic Control and Management purposes. Finally, the project *Virtual World (Virtuelle Welt)* of the DLR is presented.

4.1 Use Case: From OpenDRIVE to Shapefile

Starting with conversion of an OpenDRIVE (*.xodr*) file to ESRI Shapefile file format *.shp*, this can be obtained from the GDAL *OGR2OGR* application within the command line with the following instruction:

```
# ogr2ogr -f ''ESRI Shapefile'' output_filename.shp input_file.xodr
```

In this case, we will translate the OpenDRIVE files to Shapefiles, this is a well known vector format in the GIS community. And the $ogr2ogr - f$ command translate from any data vector format to another one: the reason, working with different data containers and vendors can be problematic. Therefore, using the standard vector representation provides *interoperability*.

For example, when the spatial analysis employs different sources of data such as OpenDRIVE and other geodata such as cadastral information, public transport networks and origin-destination (OD) matrices. The input to this system is an OpenDRIVE file, the Figure 4.1 depicts the an example of an file with the extension *.xodr* corresponding to "data_FoKr.xodr" -*Forschung Kreuzung* in Braunschweig.

```
<?xml version="1.0" standalone="yes"?>
<OpenDRIVE>
    <header revMajor="1" revMinor="2" name="" version="1.0"
        date="Fri Jun  8 15:41:02 2012" north="0.0" south="0.0" east="0.0"
        west="0.0" />
    <road name="" length="8.999999999999993" id="2" junction="-1">
        <link>
            <predecessor elementType="junction" elementId="2" />
        </link>
        <type s="0.0" type="town" />
        <planView>
            <geometry s="0.0" x="21.22164048864502" y="37.10820244328097"
                hdg="1.5707963267987355" length="8.999999999999993">
                <line />
            </geometry>
        </planView>
        <elevationProfile>
        <lanes>
        <objects>
        <signals>
            <signal s="0.0000000000000000e+00" t="5.2999999999999998e+00"
                id="9" name="" dynamic="no" orientation="-" zOffset="3.1030000090599
                type="306" country="OpenDRIVE" subtype="-1" value="-1.00000000000000
            <signal s="0.0000000000000000e+00" t="5.2999999999999998e+00"
                id="10" name="_Sg10" dynamic="yes" orientation="-"
                zOffset="0.0000000000000000e+00" type="1000001" country="OpenDRIVE"
                subtype="-1" value="-1.0000000000000000e+00">
            </signal>
        </signals>
    </road>
    <road name="" length="9.0000000000000178e+00" id="3" junction="-1">
    <road name="" length="1.8255800687128229e+01" id="4" junction="2">
    <road name="" length="1.8255800687096563e+01" id="5" junction="2">
    <road name="" length="1.8255800687241766e+01" id="6" junction="2">
    <road name="" length="9.0000000000000959e+00" id="15" junction="-1">
    <road name="" length="1.8255800687152487e+01" id="16" junction="2">
    <road name="" length="1.8255800687160701e+01" id="17" junction="2">
    <controller name="ctrl000" id="0">
        <control signalId="25" type="0" />
        <control signalId="26" type="0" />
        <control signalId="21" type="0" />
        <control signalId="22" type="0" />
```

Figure 4.1: View of an OpenDRIVE file *.xodr* in the original XML format.

The information regarding the *XODR-Driver* on the GDAL library is available once it has been compiled and configured as a driver part of the GDAL core. This type of integration allows the use of the native methods of GDAL/OGR for the OpenDRIVE files. As an illustration of the functionality of XODR in the GDAL domain the information of the driver is displayed as shown on the Figure 4.2 given the following command line instruction bellow. The detailed data obtained, summarizes the information with respect to the *XODR-Driver* as part of GDAL.

```
# ogrinfo --format xodr
```

```
D:\gdal-2.0.1_xodr\build\bin>ogrinfo.exe --format xodr
Format Details:
  Short Name: XODR
  Long Name: OpenDRIVE (XODR) driver
  Supports: Vector
  Extension: xodr
  Help Topic: drv_xodr.html
  Supports: Open() - Open existing dataset.
```

Figure 4.2: XODR in GDAL: General information of the driver.

Moreover, the properties of the OpenDRIVE files can be displayed by means of the *XODR-Driver* using the GDAL/OGR *ogrinfo* application. With the command lines given bellow, GDAL use the XODR-Driver to provide all the information regarding the dataset with *read-only* (ro) permission, including the standard version of the OpenDRIVE file.

```
# ogrinfo -ro data_FoKr.xodr
```

```
D:\gdal-2.0.1_xodr\build\bin>ogrinfo.exe -ro data_FoKr.xodr
test reading xml at xodr, getMinorRevision from xodr header:  2
 INFO: Open of `data_FoKr.xodr'
       using driver `XODR' successful.
1: data_FoKr (Multi Line String)
```

Figure 4.3: XODR in GDAL: specific information when opening OpenDRIVE files.

Another interesting option is opening the file and show all (-al) the elements contained, in terms of Simple Features. The information is read from the given input and with the *ogrinfo*, which enumerates the type of elements after the discretization and sampling processes. Figure 4.4 illustrates the results of performing the following command line instructions:

```
# ogrinfo -ro -al data_FoKr.xodr
```

```
D:\gdal-2.0.1_xodr\build\bin>ogrinfo.exe -ro -al data_FoKr.xodr
test reading xml at xodr, getMinorRevision from xodr header:  2
 INFO: Open of `data_FoKr.xodr'
       using driver `XODR' successful.

Layer name: data_FoKr
Geometry: Multi Line String
Feature Count: 16
```

Figure 4.4: XODR in GDAL: "all features" command (-al).

In addition, an script has been developed for visualizing the road network and its geometries, called *TestLineString*. This command-line program support an important tasks: exporting the *.xodr* files as ***Well Known Text*** (WKT). This capability provide the validation and testing tools of the formerly described algorithms and models. Figure 4.5 presents the results of executing the the *TestLineString*.

To sum up, one of the objectives of this Master thesis was to convert the original OpenDRIVE files given as XML text, into their accurate geometrical representation as Simple Feature. This was achieved through the implementation of an OpenDRIVE *XODR-Driver* in the GDAL/OGR domain.

Along this Use Case, it was observed the diverse potentialities to handle the OpenDRIVE geodata, now that it can configured as part of the GDAL core. Probably, one of the most relevant capabilities, is the flexibility to *adopt* other vector formats, such a container with different layout and file extension, while mantaining the precise values and attributes of the features.

A range of capabilities were enabled for the OpenDRIVE domain through the conversation of geometries to the vectorial plane by means of the *XODR-Driver*. Thus, this new functionalities allows the extension of the functions and spatial analysis over OpenDRIVE datasets such as:

- *Data collection*
- *Data storage (geo-databases)*
- *Dara management*
- *Data querying*
- *Data analysis*
- *Data presentation*
- *Dara visualization*

Subsequently, given these functions, the native GIS engines provide the following vectorial operations:

- *Aggregation*
- *Merge*
- *Simplification*
- *Collapses*
- *Amalgamation*
- *Refinement*
- *Enhancement*
- *Smoothing*
- *Exageration*
- *Displacement*

```
================================================
                  TestLineString.exe
------------------------------------------------
Usage: test_lineString.exe OpenDRIVE_datei.xodr

Author: Ana Maria Orozco
Deutsches Zentrum für Luft- und Raumfahrt e.V. DLR
================================================
GDAL Version: 2010000

Road id: 2
         s:  0.000000

Road id: 3
         s:  0.000000

Road id: 4
         s:  0.000000
         s:  1.777695
         s:  2.812178
         s:  15.443623
         s:  16.478106

              ...
Road id: 17
         s:  0.000000
         s:  1.777695
         s:  2.812178
         s:  15.443623
         s:  16.478106

 Line       (Vector size):  24
 Spiral     (Vector size):  16
 Arc        (Vector size):  8
 Poly3      (Vector size):  0
 ParamPoly3 (Vector size):  0

 Total Roads: 16
 Total geometry elements:  48
 OGR Geometry Collection Size :  32
================================================
```

Figure 4.5: *TestLineString* for validation, testing and exporting OpenDRIVE

The GIS domain offers an extensive set of geoprocessing tools, one of them is the ***QGIS*** software, a cross-platform and open-source desktop geographic information software. Moreover, there are others similar tools in the market, for example ArcGIS from ESRI and many others. With this in mind, the users of the OpenDRIVE standard will be benefit from the developed ***XODR-Driver***: bringing all the range of geotools and operations of the GIS sphere to the OpenDRIVE domain. The Figure 4.6 presents the visualization of the example data from the *Forschung Kreuzung* intersection in the City of Braunschweig, on the QGIS Environment. The typical view of geodata employs: spatial data layers layers and features, with the corresponding attributes and operations; so, OpenDRIVE data inherit these characteristics as a result of the usage of the *XODR-Driver*.

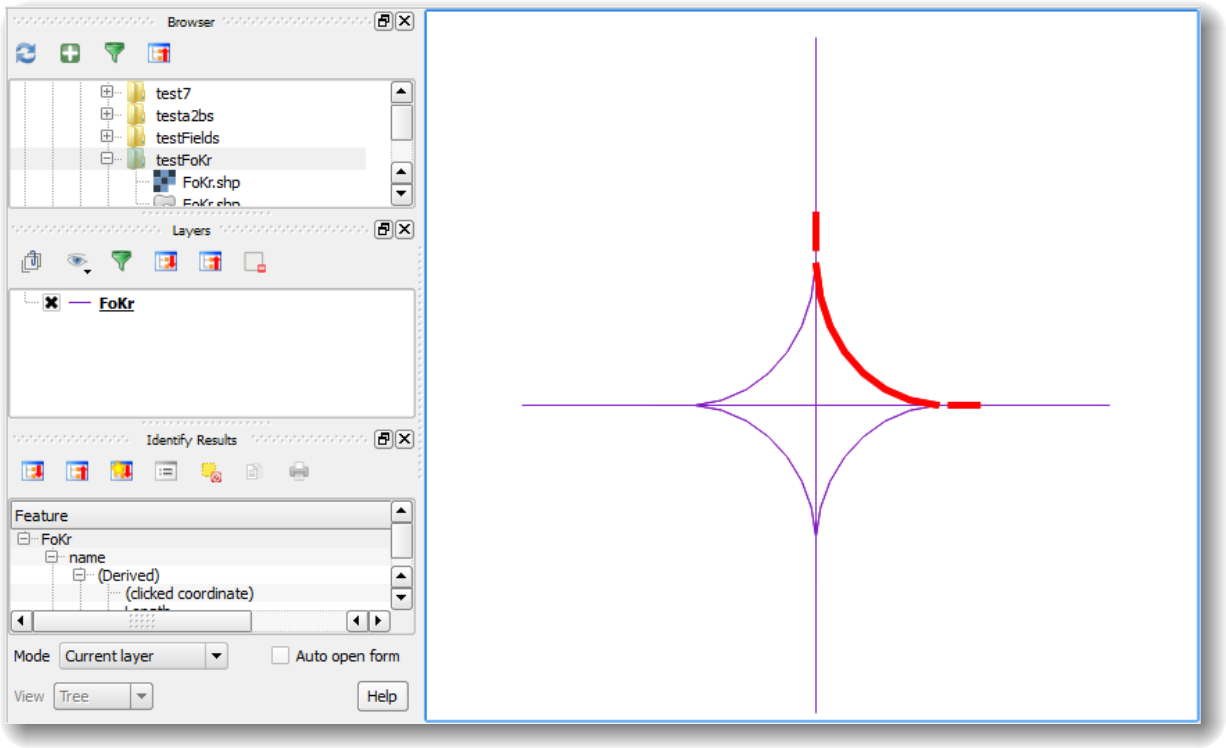


Figure 4.6: Visualization of an XODR file (Forschung Kreuzung Intersection) in QGIS

4.2 Use Case: XODR for Transportation and Traffic Engineering

As a second use case, it has been proposed the direct applicability of the OpenDRIVE standard and ***XODR-Driver*** to the area of *Traffic Management and Control*. This use case is focused on the usability of the spatial information for traffic engineering, transport planning and traffic control. In this context, the aimed users are transportation engineers which use traffic data as critical source of information. Particularly, in combination with other existing sources, such as public transport networks, cycling routes, pedestrians behavior, governmental data and so forth. The use of these spatial information in regards the mobility information, provides the context and framework for the study scenarios e.g. traffic jams, travel demand behavior, peak-hours, among others. Then, the geodata provide the accurate information and the tools to combine, merge and analyze the *thematic layers* to understand the diverse phenomena on the transportation field.

transport authorities plan routes and schedules. dynamic to meet specified objectives. monitoring transit vehicles planning

The idea behind this use case is to show the extensibility of OpenDRIVE format and the benefits of use the geodata as spatial layers by means of the *XODR-Driver*. Transport authorities can make use of OpenDRIVE for simulation purposes, which serves as a baseline for decision-making for planners, transit authorities, traffic and transportation engineers. So, the goal of this use case is to show the data matching with other sources regarding traffic control; in other words, it presents the process of merging and accumulating geodata regarding the urban traffic signalization and control.

As a start point, it is necessary to contextualize the situation regarding the use case and its application to the traffic engineering area. The data correspond to the City of Braunschweig owing to the fact that the ***Institute of Transportation*** **of the** ***German Aerospace Center*** is located in Niedersachsen, northwest of Germany, where this Master Thesis was conducted. So, the first map is given by Figure 4.7 and shows a general view of the central zone of Braunschweig and its road network. The area is around 192 km^2 with a population of 250.556 according to the City Statistics Office [Niedersachsen, 2015]. Furthermore, concerning the road network is important to mention that the main motorways are: the A2 (Berlin - Hanover - Dortmund) and the A39 (Salzgitter - Wolfsburg). As part of the study, we will focus on the intersection of two rings the *Rebering* and *Hagenring* given their importance and impact on the traffic situation.

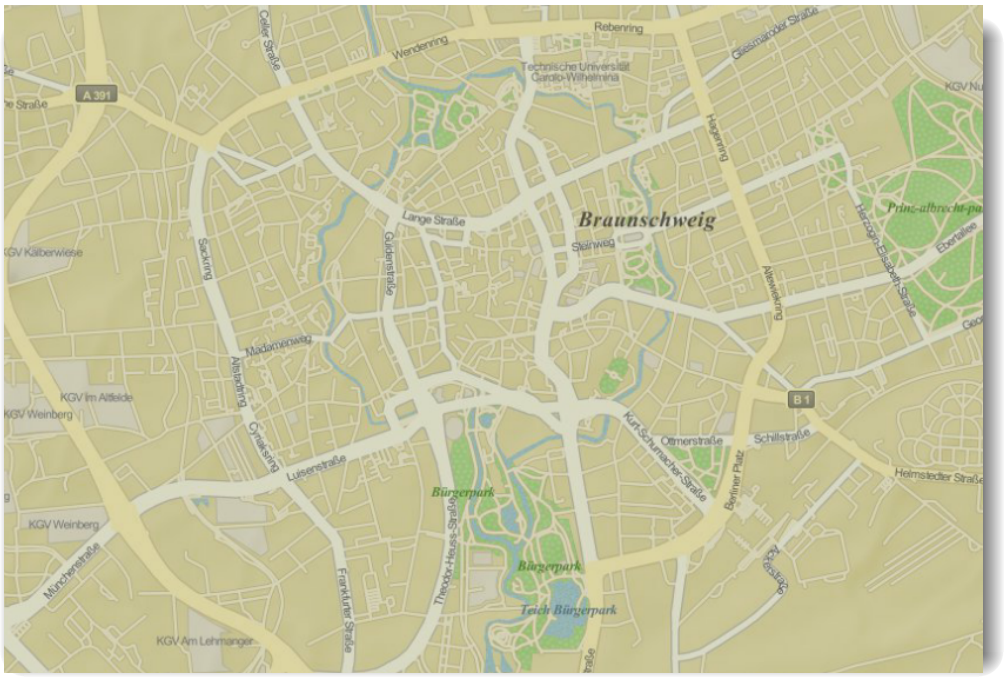


Figure 4.7: Map of the central area of Braunschweig. Source: OpenLayers and Apple Maps

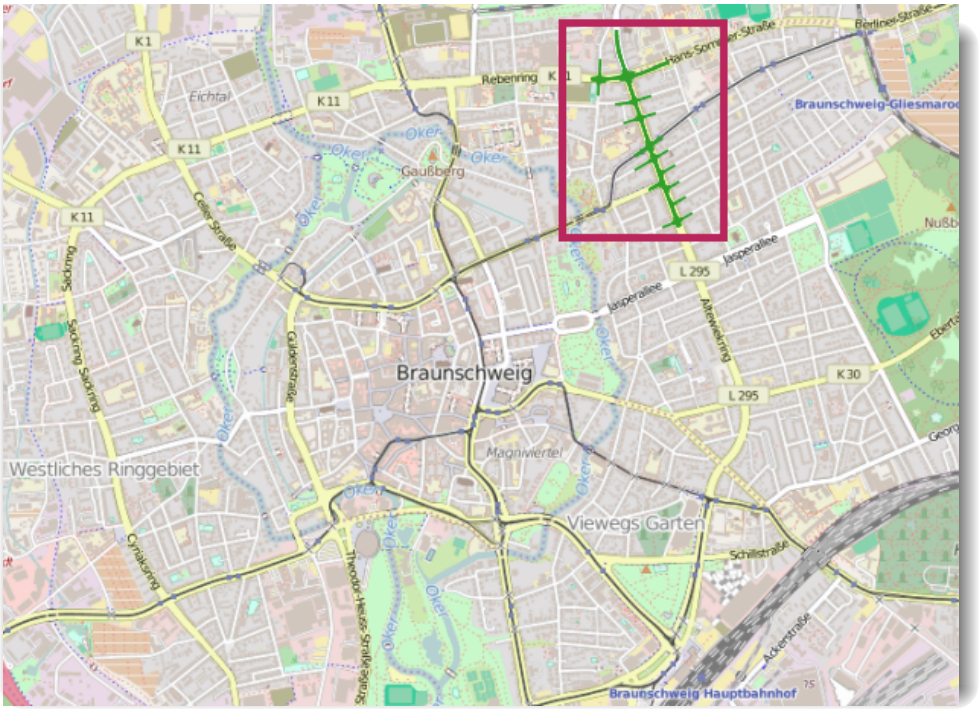


Figure 4.8: Map of the study area. Source: OpenLayers and OpenStreetMaps

(a) Pure OpenDRIVE data (b) OpenDRIVE with some OSM Polylines

Figure 4.9: Road network of one intersection in Braunschweig. Source: DLR/XODR

The first step of the process of data fusion was to convert the original OpenDRIVE file to a vectorial format. The *ESRI Shapefile* (.shp) [ESRI, 1998] was employed and by using QGIS or ArcGIS, it is possible to visualize and interact with each feature that forms this part of the network. In this case, the ***XODR-Driver*** plays the role as *geometry* or *Simple Feature factory*. As it was mentioned before, the advantage of the vector representation goes beyond from the visualization, but also allows the users to handle the data as single objects with their respective properties. In this scope, a table of attributes contains all the information of each road, with: geolocation, name, identification number or ID (for database purposes), junction and length. Moreover, for each road a set of objects and geometries describes its layout and configuration in the network. Figure 4.8 visualize the intersections and roads in the study area, using the zoom and pam functions.

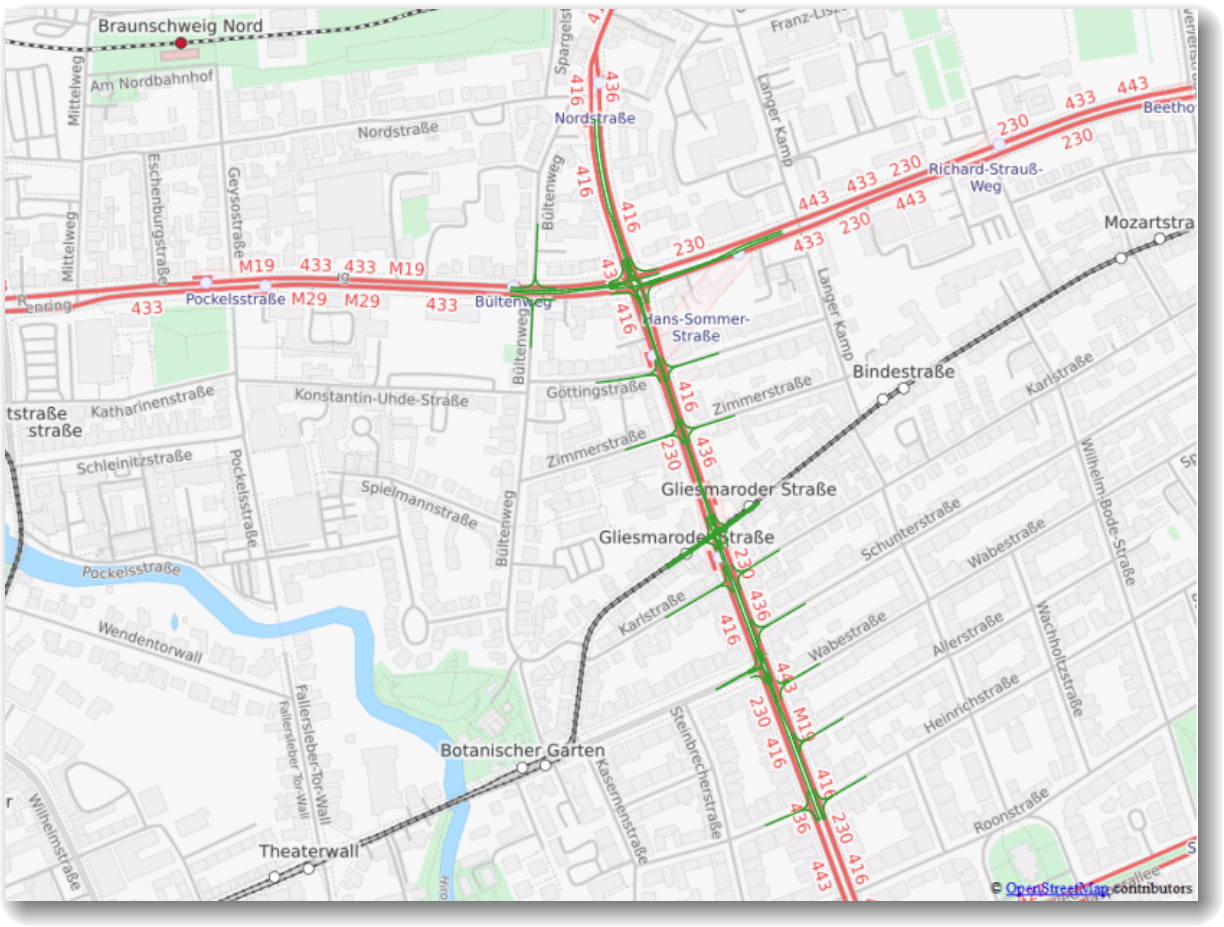


Figure 4.10: Study area with the road network from XODR and Public Transport from OpenLayers - OCM Public Transport

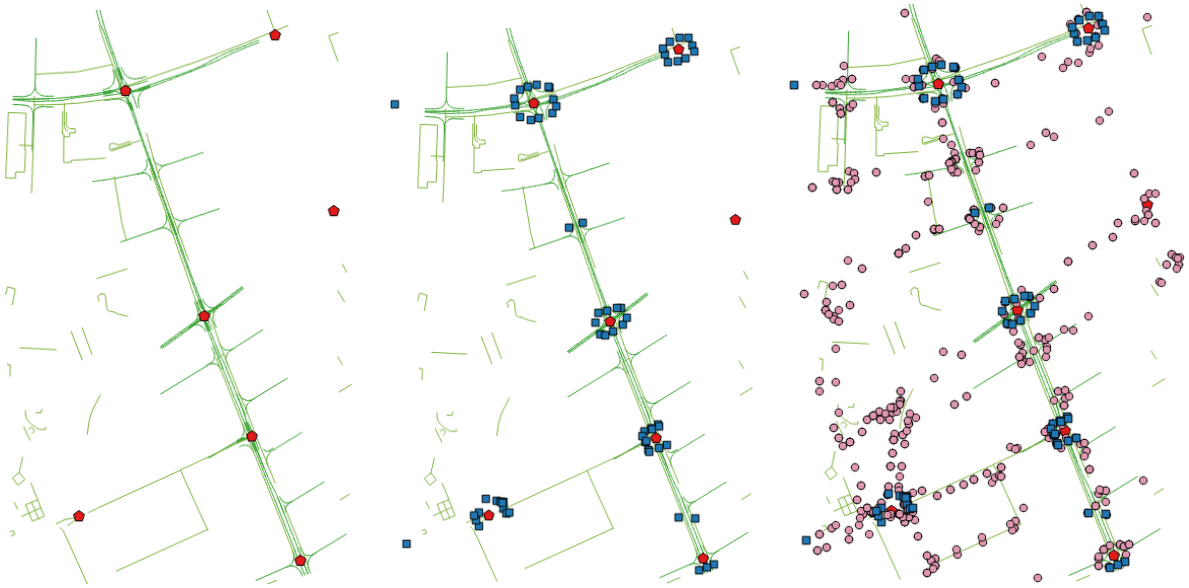
The fundamental role of the public transport networks in the decision-making regarding the planning, operations, logistics and implementation of the transportation sector, demands accurate data, mobility information and georeferenced statistics, among others. For transport professionals, the handling and operation of the diverse sources of information must be done under reliable techniques. Thus, data integrity, precision and accuracy, are requirements that any GIS software must fulfill. In this context, the *XODR-Driver* provides the precise information from the original sources and reliable GIS operations over OpenDRIVE data. In the Figure 4.10 is possible to observed that the road network (green colored lines) matches perfectly with the public transport network (red colored lines) , which denotes the precision and accuracy of the XODR-generated maps.

Figure 4.11: Error on the geo-referencing of the data

At this points a critical question rises, which are the consequences of data error or *modeling error*? To illustrate this point, the Figure 4.11 shows three overlaid maps with a significant error on the coordinate system. At any point during the setup of the geographical coordinates, there was a displacement of the actual geolocation information. Without the comparison with other sources of data, this error can pass unadvertised, and what is more problematic is that, any geographic error propagates to other thematics layers. After *debuging* the XODR-source code, the error was found and corrected. The problem was the different coordinate systems given by the version draft 1.4E and the actual released standard 1.4 of OpenDRIVE. While the the latest version use the *proj4* projection definition the previews versions use the *WGS 84* reference system. Therefore, to overcome this problem, a validation was implemented for the support all the versions of the OpenDRIVE standard.

However, not always the errors are visually perceptive, for example the numerical representation error. In these cases, it is necessary to run a set of software quality tests before releasing a functional module. When there are problems of any type, it is possible to have mismatching data on the datasets, occasioning misleadings results.

Following the same scenario and applying matching techniques for data fusion, we



(a) LSA - Traffic Lights added (b) StVO - Road Signs added (c) Signalization added

Figure 4.12: Road network of one intersection in Braunschweig. Source: DLR/XODR

will added traffic information to the XODR road network. The process consists on taking the layer of the OpenDRIVE file, and throughout *queries* on the datasets of the City of Braunschweig, provided by the DLR, it was possible to distinguish and differentiate on independent layers the information of our interest. In this case, geodata regarding **traffic light (LSA)**, **road signs (StVO)** and **signalization points (poles)**. As shown on the Figure 4.12, each of the mentioned categories is represented by a layer, and each feature is given by its geographical coordinates. The effective use of the properties and operation on the geodata permit to accumulate, complete and extend the data and its meaning. These other uses enriched the studied phenomena, since the users are able to link informations to their own geodata, not only from different sources but from other thematics models. A common example is the field of public transport, which take in count demographical and cartographical studies for the transport demand management. On these series of images, the sources are provided in different data formats and vendors. In Braunschweig, Bellis [BELLIS, 2015] manages the data of LSA, StvO and poles. According to [BELLIS, 2015]: "BELLIS operates traffic lights, and is responsible for parking management, traffic management, road signs and markings. BELLIS plans and organises traffic control measures, as well as traffic backups at construction projects and events".

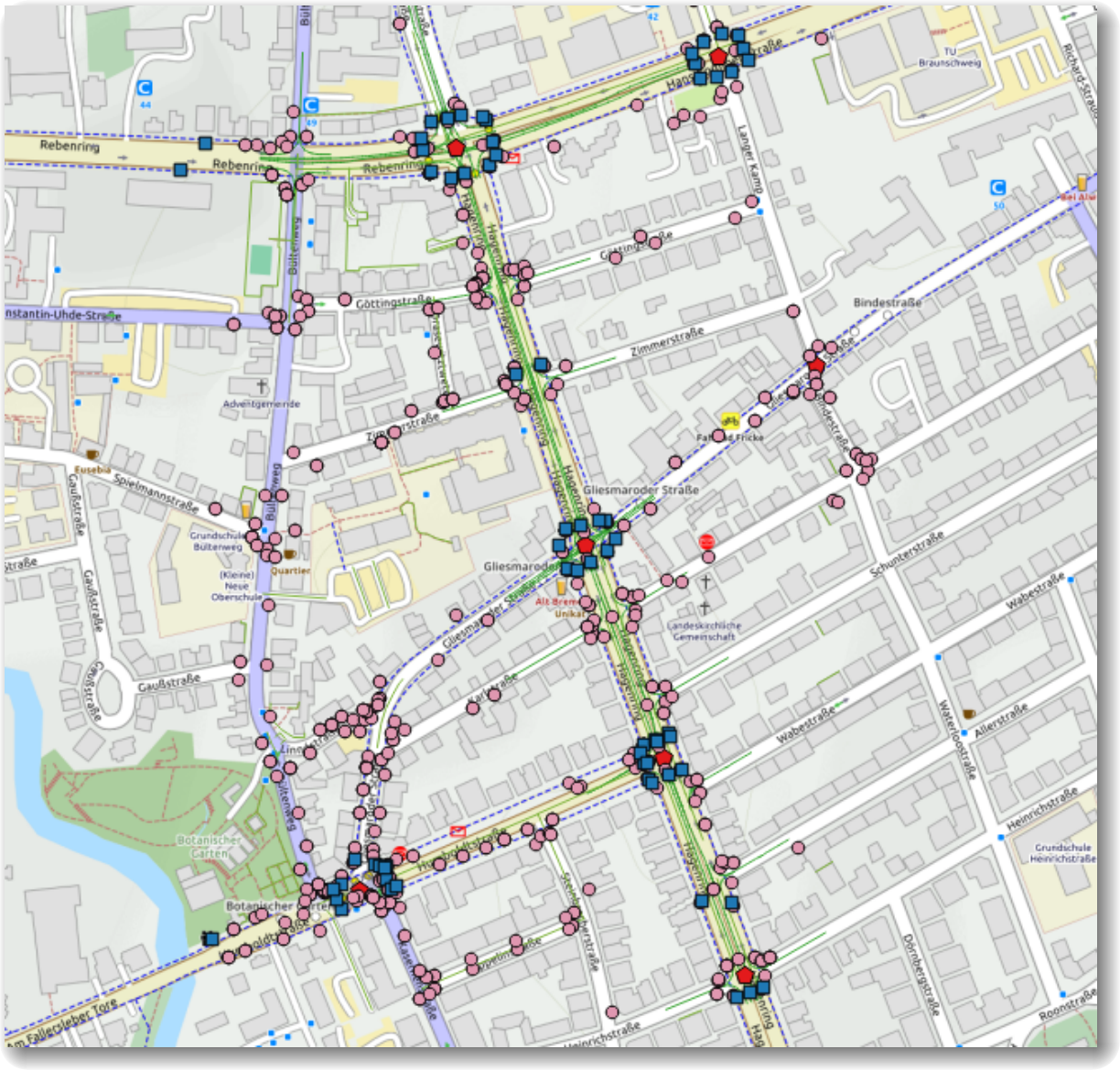


Figure 4.13: Data merging of Road Networks and Traffic Signalization

As depicted on Figure 4.15, the merging techniques of geodata in combination with other capabilities like ***re-mapping***, ***layers overlaying***, ***data measuring***, ***changing the scale: zoom and pan***, among others, are the fundamental geodata operations. The *XODR-Driver* enables a geotoolbox for the user of OpenDRIVE to enhance, analyze and management the data; this allows to ***handle***, ***create***, ***share***, ***map***, ***update*** and ***maintain*** spatial and geographical-based models from the OpenDRIVE standard.

4.3 OpenDRIVE in Driving Simulation

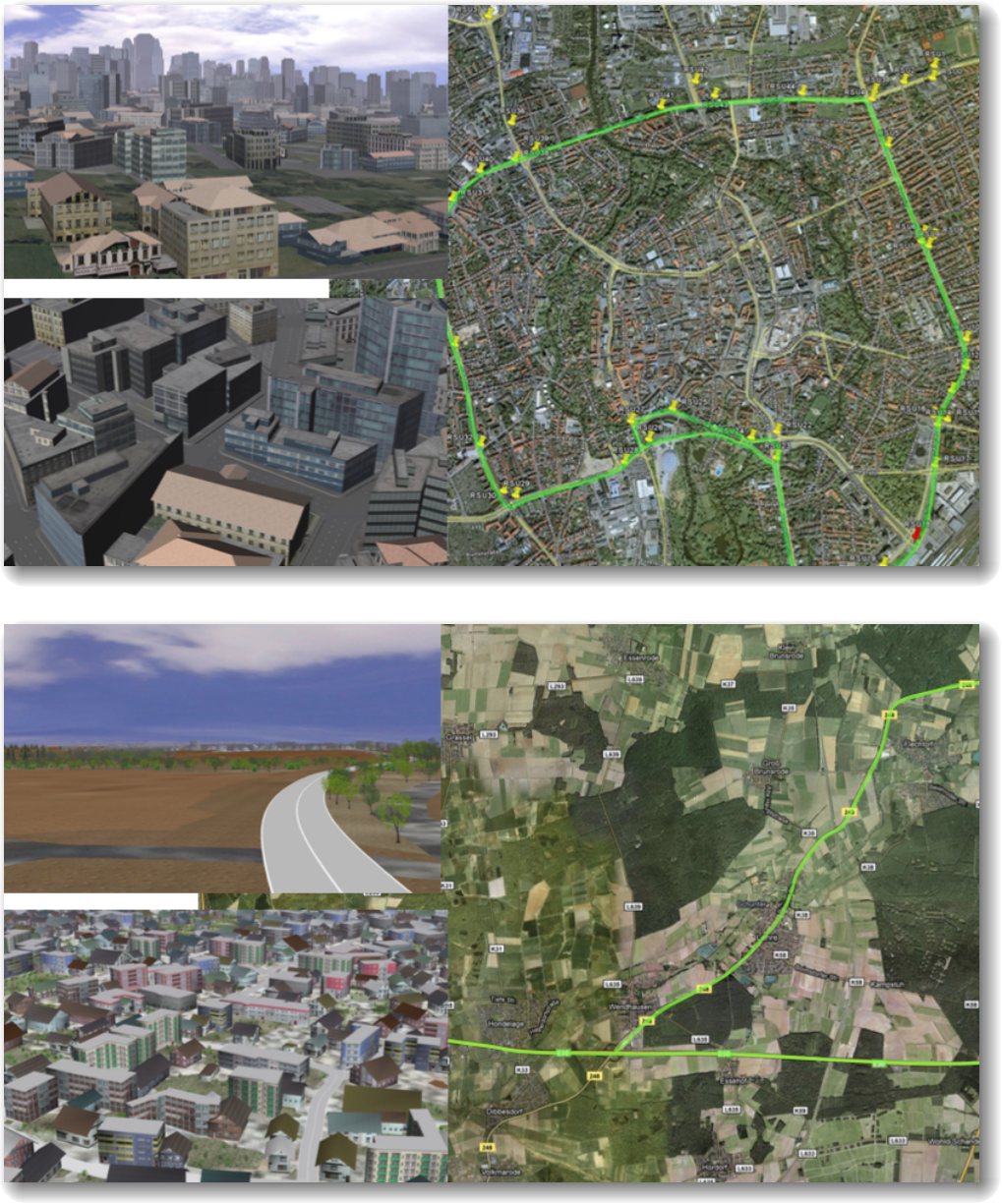


Figure 4.14: 3D City Model (Braunschweig) generated from diverse spatial data source, including the OpenDRIVE standard. Source: DLR

"The interdisciplinary project $SimWorld^{URBAN}$ - initiated by the German Aerospace Center (DLR) - aims to improve and to facilitate the generation of virtual landscapes for driving simulators. It integrates the expertise of different research institutes working in the field of car simulation and remote sensing technology. SimWorld will provide detailed

virtual copies of the real world derived from air- and satellite-borne remote sensing data, using automated geo-scientific analysis techniques for more efficiency and greater realism for landscape models. The implementation of geo-databases and GIS technology within the simulator will allow for further simulation and testing of new technologies like e.g. radar-sensors, night vision systems as well as positioning systems such as GPS and Galileo." [Sparwasser et al., 2010].

In the context of the Car-to-X (C2C and C2I) communication and driver assistance systems the simulation plays an important role. The use of realistic urban scenarios are essential on the design, model and implementation of these systems. The project *Virtual World* of the DLR, creates a digital atlas, for the representation of multi-modal metropolitan regions considering the transport infrastructure (roads, railways, buildings, environment, and so forth) [Richter and Friedl, 2015]. In this case, the OpenDRIVE standard provide the road network description for driving simulation. In the *Virtual World* project the the logical street descriptions generation is automated and the data completion is conducted with the available spatial data (cadastre data, OpenStreetMaps, Navteq) [Scholz, 2014].

Figure 4.15: Driving Simulators and Virtual Reality-Lab at DLR. Source: DLR

Chapter 5

Conclusions and Outlook

In this work, it has been proposed the conceptualization and software development of the ***OpenDRIVE XODR-Driver***: from the analytical representation of the roads geometries to the definition of features on the Simple Feature model. Moreover, a new driver format has been developed and integrated with the geospatial library GDAL. As a result, it extended the operability of the OpenDRIVE datasets with other formats as well.

This master thesis combines the fundamental concepts of Informatics, Transportation Systems and Geographic Information systems to model, develop and implement the OpenDRIVE *XODR-Driver*. During the construction of the mathematical representation and its subsequently code implementation novel design challenges were stated. A solid baseline on linear algebra, euclidean geometry and vectorial mathematics were pre-requisites to model the roads as vector. As well as the deep understanding of the C++ programming language and the object-oriented paradigm played a key role.

The robustness and flexibility of the software architecture of OpenDRIVE *XODR-Driver* lead to present as the result of this master thesis, the first functional prototype of the *XODR-Driver*. In addition, efficient and elegant programming techniques were used to build a high performance software solution. The use of the driver is extensible to other areas: like transport planning and traffic control, realistic traffic and driving simulation, navigation and assisted driving application, and other further application on the geoinformatics and ITS scopes.

In the transportation field, the applicability of certain spatial information requires the the geoprocessing tools for the accurate management of the geodata. Nowadays a broad range of application employed geo-referenced data as an illustration, some geolocated-service to mention are: route guidance (GPS), optimization of road networks whether

for private or public transport, the simulation of adaptive control measure and its further implementation, and many others. This thesis had provided a software solution based on a discrete mathematical model, which is capable to generate a road network layout and the components along, in the standard vectorial plane from the OpenDRIVE format.

Future Work

For the future work, the remaining set of elements (e.g. stations, user-defined objects) and geometry components (i.e. spiral and polynoms) of the roads are to be completed. For the case of spiral the euclidian form is an optimal way to describe the continuous function and subsequently for the discretization to the vectorial domain. The spirals and polynoms will provide the transition curves between arcs and straight sectors of the road. This, in order to smooth the curvature.

Another interesting open issue is, the representation of dynamic traffic control elements. At this point, the question would be how to include the user-defined tags and objects on the files. Or even more interesting, how could OpenDRIVE be linked with other traffic control simulators e.g. Sitraffic (Siemens), PTV Vissim, etc. This feature can provide more realism at intersections and junctions on the driving simulator, even, the possibility of *detection mechanisms* e.g. inductive loops on the road. Also, the real waiting times for pedestrians, circle times, green and red time and the interaction with the coordination (green waves) and signal plans could be included. By now, the Simulation of Urban MObility (SUMO) [Behrisch et al., 2011] supports the OpenDRIVE standard.

On the other hand, regarding the software component, it has been observed during this investigation that the integration to GDAL, is actually complicated to carry on. Because of that, it has been proposed as future work, a simpler way to integrated *XODR-Driver* to GDAL/OGR. The *Plug-and-Play* philosophy has been proposed as an alternative method to make use the *XODR-Driver*, hence the continuity of this project is going towards the development of an external *Plugin*. This solution does not require the compilation of the core code neither the additional configuration in GDAL. This aims is to contribute to the simulation community and bring the developed *XODR-Driver* as a ready-to-use software module and its eventual release in the open source community.

Bibliography

Adams, R. A. (1991). *Calculus*. Addison-Wesley.

Behrisch, M., Bieker, L., Erdmann, J., and Krajzewicz, D. (2011). SUMO - Simulation of Urban MObility. In *The Third International Conference on Advances in System Simulation (SIMUL 2011), Barcelona, Spain*.

BELLIS (2015). Bellis gmbh, braunschweig. *Retrieved October 20, 2015, from http://www.bellis.de*.

Booch, G., Maksimchuk, R. A., Engle, M. W., Young, B. J., Connallen, J., and Houston, K. A. (2008). Object-oriented analysis and design with applications. *ACM SIGSOFT software engineering notes*, 33(5):11.

Chang, K.-t. (2006). *Introduction to geographic information systems*. McGraw-Hill Higher Education Boston.

CodeSynthesis (2014). XSD: XML Data Binding for C++. *Retrieved November 24, 2015, from http://www.codesynthesis.com*.

Dupuis, M. and et. al. (2015). Opendrive format specification, rev. 1.4, issue h. *OpenDRIVE Project. http://www.opendrive.org*. Issuing Party: VIRES Simulationstechnologie GmbH.

ESRI, E. (1998). Shapefile technical description. *An ESRI White Paper*.

Haklay, M. and Weber, P. (2008). Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE*, 7(4):12–18.

Haubrich, T. (2013). *Konzeption einer Verkehrsnetzrepräsentation für kognitive Agenten in virtuellen Umgebungen*. Hochschule Bonn-Rhein-Sieg.

Kurteanu, D. and Kurteanu, E. (2010). Open-source road generation and editing software. *rapport*, (2010).

Larman, C. and Basili, V. R. (2003). Iterative and incremental development: A brief history. *Computer*, (6):47–56.

Ljung, L. (1998). *System identification*. Springer.

Longley, P. A., Goodchild, M. F., Maguire, D. J., and Rhind, D. W. (2001). Geographic information system and science. *England: John Wiley & Sons, Ltd*.

Meyers, S. (2014). *Effective Modern C++: 42 Specific Ways to Improve Your Use of C++ 11 and C++ 14*. OReilly Media, Inc.

Miller, H. J. and Shaw, S.-L. (2001). *Geographic Information Systems for transportation: principles and applications*. Oxford University Press on Demand.

Mutschler III, E. O. and Stefaniak, J. P. (1999). Method for extending the hypertext markup language (html) to support enterprise application data binding. US Patent 5,940,075.

Niedersachsen, S. (2015). City of braunschweig, city statistics office. *Retrieved November 10, 2015, from http://www.braunschweig.de*.

Obe, R. and Hsu, L. (2011). *PostGIS in action*. Manning Publications Co.

Öberg, K. (2012). Horn-hank and opendrive road networks: An editor for creating hank scenarios while working with opendrive.

Pierce, B. C. (2002). *Types and programming languages*. MIT press.

Richter, A. and Friedl, H. (2015). Parametrisierte stadtmodelle fr fahrsimulatoren. *GIS Talk 2015, 19.-21. Mai 2015, Unterschleissheim, Deutschland.*

Rouault, E. (2015). GDAL - Geospatial Data Abstraction Library (documentation). *GDAL/OGR 2.0.1 Released. Retrieved November 24, 2015, from http://www.gdal.org/.*

Scholz, M. (2014). Vom kataster in die fahrsimulation: Opendrive aus geodaten. *Next Generation Forum 2014, 20. Okt. 2014, Göttingen.*

Seele, S., Herpers, R., Bauckhage, C., and Becker, P. (2012). Cognitive agents with psychological personality profiles for traffic simulations in virtual environments. *Sommertreffen Verkehrssimulation 2012*, page 8.

Shi, H. (2011). Automatic generation of opendrive roads from road measurements.

Sparwasser, N., Friedl, H., Krau, T., Meisner, R., and Stobe, M. (2010). Simworld: Automatic generation of realistic landscape models for real time simulation environments–a remote sensing and gis-data based processing chain. *Advances in transportation studies*, (21):15–22.

Taylor, J. R. and Cohen, E. (1998). An introduction to error analysis: the study of uncertainties in physical measurements. *Measurement Science and Technology*, 9(6):1015.

VIRES (2015). OpenDRIVE. *VIRES Simulationstechnologie GmbH, OpenDRIVE Project. Retrieved November 26, 2015, from http://www.opendrive.org.*

Xerces (2015). Apache software foundation. *Retrieved November 24, 2015, from http://xerces.apache.org.*

Zimmermann, A. (2012). *Basismodelle der Geoinformatik: Strukturen, Algorithmen und Programmierbeispiele in Java.* Carl Hanser Verlag GmbH Co KG.

List of Figures

List of Tables

Acknowledgment

To my beloved family.
To my friends, who are walking this path with me.
And to God.

Masters Thesis Declaration

Declaration concerning the Masters Thesis

I hereby confirm that the presented thesis work has been done independently and using only the sources and resources as are listed. This thesis has not previously been submitted elsewhere for purposes of assessment.

Munich, December 11th, 2015

Ana Maria Orozco I.	Date