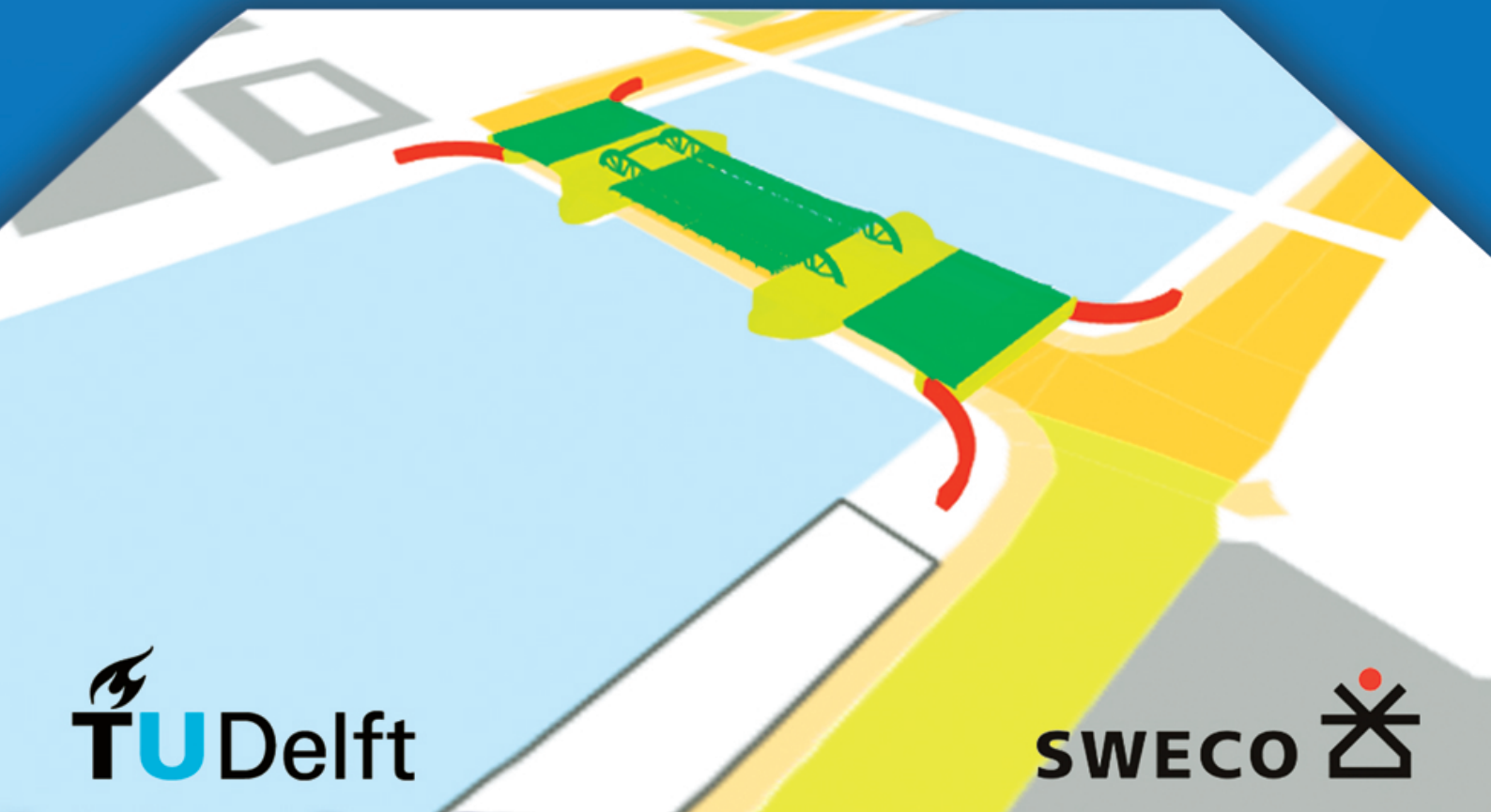
MSc thesis in Geomatics for the Built Environment

Integration of 3D BIM Models in a Web GIS for Life Cycle Asset Management

Manuela Manolova

2018



TU Delft

SWECO

INTEGRATION OF 3D BIM MODELS IN A WEB GIS FOR LIFE CYCLE ASSET MANAGEMENT

A thesis submitted to the Delft University of Technology in partial fulfillment
of the requirements for the degree of

Master of Science in Geomatics for the Built Environment

by

Manuela Manolova

November 2018

Manuela Manolova: *Integration of 3D BIM Models in a Web GIS for Life Cycle Asset Management* (2018)
This work is licensed under a Creative Commons Attribution 4.0 International License. To view a copy of this license, visit
http://creativecommons.org/licenses/by/4.0/.

ISBN 999–99–9999–999–9

The work in this thesis was made in the:



GIS Technology Group
Department of the OTB
Faculty of Architecture & the Built Environment
Delft University of Technology



OBSURV Software Team
Department of GIS and ICT
Division of Transportation and Mobility
Sweco Netherlands

Thesis Committee:

Drs. Marianne de Vries
First Supervisor

Prof. mr. dr. Hendrik Ploeger
Second Supervisor

Dr. ir. Bastiaan van Loenen
Co-Reader

Hilbert Davelaar
First Company Supervisor

Henri Veldhuis
Second Company Supervisor

CONFIDENTIALITY CLAUSE

The thesis is based on internal, confidential data and information of Sweco Netherlands.
The author's rights on the achieved results lie with the host organisation.

ABSTRACT

In recent years, there has been a growing necessity for 3D geoinformation in urban planning and infrastructure management to provide a more realistic representation of urban areas and the built environment. A key concept in public infrastructure management is Life Cycle Asset Management (LCAM), which aims to improve the decision making in each phase of the life cycle of infrastructure assets, such as road and utility networks, civil structures, and green areas. Traditional systems in LCAM are limited in terms of analysis and visualization opportunities due to the use of simpler tools like tables. In this respect, Geographic Information System (GIS) solutions have been widely integrated in LCAM to optimise the administration and monitoring of infrastructure assets.

The incursion of semantic 3D models complements the use of GIS by introducing new and innovative modelling and visualization methods. As one of the most common design techniques in the building and construction industry, Building Information Modelling (BIM) facilitates the generation of multidimensional models to describe the physical and functional characteristics of built objects. This research studies the integration of 3D BIM models in a Web GIS system for public space management to improve the procedure in the maintenance phase of LCAM. The goal is to solve the challenges related to the acquisition, processing, and visualization of 3D models.

The thesis outlines the process of the development of a 3D prototype in an attempt to demonstrate a possible solution for the integration of BIM models in an existing GIS application. The research starts with a scientific literature review to familiarise with the LCAM concept, Web GIS, and semantic 3D models. Then, the stakeholders in the project are identified and their requirements with regard to the functional capabilities of the prototype are gathered in order to create a design concept. Consequently, the BIM model used in this study is presented and processed to comply with the Dutch national standards for inspection and maintenance. The prototype is developed on the basis of Web Graphics Library (WebGL), a technology for web rendering of interactive 3D graphics and animations supported by all modern web browsers. At the end of the development process, the prototype is evaluated by experts in the LCAM and BIM fields to assess its usability by means of a questionnaire survey. The input from the participants in the survey is essential for the further improvement of the prototype and its release as fully functional product.

ACKNOWLEDGEMENTS

I would like to express my appreciation to the people who played a major role in the the completion of this project.

First, I would particularly like to thank Marianne de Vries for her supervision, all the brainstorming sessions, and great technical support for the prototype development. She was always ready to help me when I ran into troubles or had questions regarding the research. I would also like to acknowledge Hendrik Ploeger as second supervisor for his valuable advice on the theoretical aspects of the thesis. Moreover, I am grateful to Bastiaan van Loenen as for his critical and constructive comments on the thesis as co-reader.

I would like to express my sincere gratitude to Hilbert Davelaar for supporting and guiding me through my graduation project. A big thanks goes to Henri Veldhuis for always giving me new ideas how to improve the project and organising meetings with different specialists in the field of the research.

Furthermore, special thanks to all my colleagues, particularly Marijn van den Berg, Ivan Klop, Marco Sprengelmeijer, Kevin Thyssen, Myron Ramkisoen, and Rony Nedkov, for their cooperative support and help during the period of my research.

In addition, I want to thank the experts from the Municipality of Rotterdam who agreed to participate in the project and shared their knowledge and expectations with me: Jan de Jong, Joris Goos, Helmer Heijden, and Christian Veldhuis.

Last but not least, I would like to express my deepest appreciation to my family and friends for their support and continuous encouragement in the two years of my study.

CONTENTS

1 INTRODUCTION 1
1.1 Problem Statement . . . 1
1.2 Research Objective . . . 2
1.3 Thesis Outline . . . 3
2 THEORETICAL BACKGROUND 5
2.1 Life Cycle Asset Management . . . 5
2.2 Web GIS . . . 8
2.3 Semantic 3D Models . . . 11
2.3.1 BIM . . . 11
2.3.2 CityGML . . . 14
2.4 Related work . . . 15
3 PROTOTYPE DESIGN 21
3.1 Requirements Specification . . . 21
3.2 User Interface Design . . . 23
3.3 WebGL Frameworks . . . 26
3.3.1 Three.js . . . 27
3.3.2 iTowns . . . 27
3.3.3 Cesium . . . 28
3.3.4 OSM Buildings . . . 28
3.4 Frameworks Comparison . . . 29
4 BIM MODEL PROCESSING 31
4.1 BIM Dataset . . . 31
4.2 BIM Decomposition . . . 32
5 PROTOTYPE DEVELOPMENT 37
5.1 Test Environment Preparation . . . 37
5.2 3D Viewer Configuration . . . 40
5.2.1 Scene Creation . . . 40
5.2.2 BIM Model Loading . . . 41
5.2.3 Base Map Configuration . . . 44
5.2.4 Object Selection . . . 52
5.2.5 Additional Functionalities . . . 55
6 PROTOTYPE EVALUATION 59
6.1 Usability Testing . . . 59
6.2 Survey Research . . . 60
6.3 Discussion . . . 66
7 CONCLUSIONS AND FUTURE WORK 69
7.1 Conclusions . . . 69
7.2 Contribution to Geomatics . . . 72
7.3 Future Work . . . 72
A BIM MODEL COMPOSITION 83
B BIM MODEL DECOMPOSITION 85
C UML DIAGRAM OF THE DATABASE STRUCTURE 87
D HTML, CSS AND JAVASCRIPT CODES 89

E PL/SQL CODE 113
F QUESTIONNAIRE SURVEY 115
G REFLECTION 121

LIST OF FIGURES

Figure 1.1 Process of the prototype development. 3
Figure 2.1 Life cycle of infrastructure assets, adapted from Too [2008]. . . . 5
Figure 2.2 NEN levels for the infrastructure assets, adapted from Boanca [2014]. 6
Figure 2.3 Client-server architecture, adapted from Alesheikh et al. [2002]. 8
Figure 2.4 OGC standards for web mapping [Baumann and Aiordachioaie, 2009]. 9
Figure 2.5 Multi-tier client-server architecture of OBSURV, adapted from Tepas [2014]. 10
Figure 2.6 LOD in the BIM technology specified by BIMForum [2016]. . . . 12
Figure 2.7 BIM implementation in the building life cycle [Dispenza, 2010]. 12
Figure 2.8 LOD in CityGML [Biljecki et al., 2017]. 14
Figure 2.9 UML class diagram of the semantic and geometric properties for the building model [Kolbe, 2009]. 15
Figure 2.10 3D city model of Berlin visualised in Cesium by Schilling et al. [2016]. 16
Figure 2.11 3D city models of Rotterdam created by Prandi et al. [2015] in: (a) NWW applet. (b) Cesium. 16
Figure 2.12 Web platform for 3D buffer analysis developed by Chaturvedi [2014]. 17
Figure 2.13 BIM model of a bridge integrated in OBSURV by Boanca [2014]. 18
Figure 2.14 Georeferenced BIM model of a building in The Hague [Diakite, 2018]. 18
Figure 2.15 Georeferenced 3D model of a building by Kolár and Wen [2009]. 19
Figure 2.16 Meteorological model with integrated Google Earth platform created by Wang et al. [2013]. 19
Figure 3.1 Mockup of the UI design of the 3D prototype. 25
Figure 3.2 Software architecture of dynamic web pages and web pages using WebGL technology, adapted from Matsuda and Lea [2013]. . 26
Figure 3.3 Customised application for city car driving built on Three.js [Poppe, 2017]. 27
Figure 3.4 Application for extruded WFS buildings on a virtual globe based on iTowns [iTowns, 2017]. 28
Figure 3.5 CyberCity 3D buildings visualized in Cesium for usage in planning and analysis, real estate, economic development, and flood risk mitigation [Cesium, 2018]. 28
Figure 3.6 3D buildings visualized in OSM Buildings [OSM Buildings, 2018]. 29
Figure 4.1 3D model of Koninginnebrug. 31
Figure 4.2 Composition of the in the initial 3D model in Autodesk Inventor. 32
Figure 4.3 Decomposition of the 3D model according to the NEN 2767-4 standards. 33
Figure 4.4 Splitting of the abutment part. 34
Figure 4.5 Decomposed assembly of the bridge abutment. 34
Figure 4.6 Span lock components in the bridge deck composition. 34

Figure 4.7 Pivoting point composition. 35
Figure 4.8 Bridge decomposition: (a) Retaining structure. (b) Substructure. (c) Superstructure. (d) Span locks. 35
Figure 5.1 Configuration of interactive report of civil constructions. 38
Figure 5.2 Home page of the custom module. 38
Figure 5.3 Rendering process in Three.js, adapted from Taalman [2015]. . . 40
Figure 5.4 Perspective camera model, adapted from Dirksen [2013]. 40
Figure 5.5 Data retrieval by using AJAX calls in APEX. 42
Figure 5.6 3D viewer with the integrated BIM model and legend. 44
Figure 5.7 Image of the area of interest, obtained as a WMS request. . . . 45
Figure 5.8 Cartesian CRSs: (a) Traditional reference system. (b) UCS reference system. 46
Figure 5.9 Reference points and their coordinates obtained from Google Earth. 47
Figure 5.10 Schematic representation of the WGS84 and ECEF coordinates [Sanz Subirana et al., 2011a]. 48
Figure 5.11 Schematic representation of the ECEF and ENU coordinates, adapted from Sanz Subirana et al. [2011a]. 48
Figure 5.12 BIM model transformation. 50
Figure 5.13 Transformation of the BIM model in CloudCompare: (a) Translation of the model to fit in the image. (b) Manual rotation of 3D model to align with the base layer. 51
Figure 5.14 BIM model transformation obtained from CloudCompare. 52
Figure 5.15 Ray casting method implemented in Three.js, adapted from Dirksen [2013]. 52
Figure 5.16 Object selection and highlight functionality. 54
Figure 5.17 Specification of the mouse position in the HTML document and the window coordinates in the 3D viewer, adapted from Kantor [2018]. 54
Figure 5.18 Object selection and highlight functionality with fixed mouse position. 55
Figure 5.19 Bridge substructure and span locks in combination with the rotate and zoom functions. 55
Figure 5.20 Change detection functionality in the 3D viewer. 56
Figure 5.21 Resized WebGL renderer and 3D model. 57
Figure 6.1 Steps in the survey research process, adapted from Neuman [2014]. 60
Figure 6.2 Pie chart showing the familiarity of the participants with OBSURV. 63
Figure 6.3 Bar chart showing the grades of the 3D viewer functionalities given by the participants. 64
Figure 6.4 Pie chart showing the most useful functionality of the 3D viewer according to the participants. 65

LIST OF TABLES

Table 3.1 Interviewees participated in the graduation project. 22
Table 3.2 Comparison of the existing WebGL frameworks for 3D web visualization. 29
Table 6.1 Occupation field of the participants in the survey. 63
Table 6.2 Calculation of the overall score of the functionalities of the 3D prototype. 64

LIST OF ALGORITHMS

Algorithm 1 Configure collapsible page 39
Algorithm 2 Associate 2D objects with their corresponding 3D model 39
Algorithm 3 Configure camera, lights, and animation controls 41
Algorithm 4 Load the OBJ files in the scene 42
Algorithm 5 Configure the PL/SQL procedure to retrieve data from Oracle . 43
Algorithm 6 Retrieve data from database and add it to the model properties 43
Algorithm 7 Assign initial colour to the model elements 44
Algorithm 8 Determine bounding box of the ground plane 45
Algorithm 9 Create ground plane to place base map 46
Algorithm 10 Convert coordinates with Three.js 49
Algorithm 11 Construct transformation matrix and apply it to the 3D model . 50
Algorithm 12 Apply transformation matrix generated by CloudCompare to the 3D model 51
Algorithm 13 Configure object selection and highlight functionality 53
Algorithm 14 Configure switcher functionality 55
Algorithm 15 Configure change detection functionality 56
Algorithm 16 Make WebGL renderer resizable 57

ACRONYMS

3DCityDB	3D City Database ... 15
AEC	Architecture, Engineering and Construction ... 11
AIA	American Institute of Architects ... 11
AJAX	Asynchronous JavaScript and XML ... 42
AM	Asset Management ... 1
APEX	Application Express ... 10
API	Application Programming Interface ... 27
AR	Augmented Reality ... 27
B3DM	Batched 3D Model ... 16
BIM	Building Information Modelling ... vii
CAD	Computer Aided Design ... 12
CRS	Coordinate Reference System ... 19
CSS	Cascading Style Sheets ... 37
DBMS	database management system ... 8
DTM	Digital Terrain Model ... 14
ECEF	Earth-Centered, Earth-Fixed ... 47
ENU	East, North, Up ... 48
GII	Geographic Information Infrastructure ... 8
GIS	Geographic Information System ... vii
GLSL ES	OpenGL Shading Language Embedded Systems ... 26
glTF	GL Transmission Format ... 16
GML	Geography Markup Language ... 9
GPS	Global Positioning System ... 47
HTML	HyperText Markup Language ... 26
HTTP	HyperText Transfer Protocol ... 8
IFC	Industry Foundation Classes ... 13
IGES	Initial Graphics Exchange Specification ... 13
IGN	Institut National de l'information Géographique et Forestière ... 27
IPT	Inventor Part ... 34
ISO	International Organisation for Standardisation ... 31
JSON	JavaScript Object Notation ... 27
KML	Keyhole Markup Language ... 17
LCAM	Life Cycle Asset Management ... vii
LOD	level of detail ... 14
LOD	level of development ... 11

NWW	NASA World Wind	16
OGC	Open Geospatial Consortium	8
OpenGL	Open Graphics Library	26
OSM	OpenStreetMap	28
PDOK	Publieke Dienstverlening Op de Kaart	10
PL/SQL	Procedural Language/Structured Query Language	10
RDP	Remote Desktop Protocol	11
STEP	Standard for the Exchange of Product Data	13
STL	Standard Tessellation Language	13
SQL	Structured Query Language	10
SVG	Scalable Vector Graphics	9
UCS	User Coordinate System	46
UI	user interface	3
UX	user experience	23
UCD	user centered design	21
URL	Uniform Resource Locator	9
VR	Virtual Reality	24
WCS	Web Coverage Service	8
WebGL	Web Graphics Library	vii
WFS	Web Feature Service	8
WGS84	World Geodetic System 1984	47
WMS	Web Map Service	8
XML	eXtensible Markup Language	30

1 INTRODUCTION

This thesis provides a framework for the integration of 3D BIM models in a Web GIS system for public space management. This topic is relatively new in the Geomatics field, but widely discussed due to its potential to improve the procedures in urban planning and infrastructure management. The key concepts covered in this research are LCAM and its stages, Web GIS systems, and semantic 3D models.

1.1 PROBLEM STATEMENT

Public infrastructure comprises road and utility networks, civil constructions, green areas, sport fields, and underground pipelines, which are owned by public organisations and used by the citizens. These assets are characterised by relatively long service life and require a regulated approach for their long-term management. In this respect, Asset Management (AM) includes a range of strategies for keeping a systematic record of the individual assets, planning maintenance, and implementing appropriate information systems to facilitate the entire management process [Cagle, 2003]. LCAM is one of the strategies, which represents an integrated approach for improving the decision making in each phase of the asset life cycle, from planning and design through building, maintenance and operation, to disposal and recycling [Wolter, 2010].

In the last few years, GIS systems have been extensively used for modelling, analysing and visualizing urban areas and the built environment [Scianna, 2013]. The advances in web technology open up new opportunities for GIS by providing a variety of services for geospatial data [Tatarević, 2007]. Web GIS enables easier access, sharing and exchange of geoinformation through standard web protocols. Therefore, the Web GIS framework is utilised in a wide range of fields, such as urban planning, architecture, construction, and infrastructure management [Jeberson Retna Raj and Sasipraba, 2010].

Lately, there has been a growing interest in 3D GIS systems to provide more realistic representation of real-world phenomena and allow for advanced spatial analysis. However, there are many complications in the development of sophisticated 3D solutions related to the complexity of the surface and subsurface geometries [Abdul-Rahman and Pilouk, 2008].

Along with the development of 3D GIS, several LCAM phases are moving towards 3D modelling of complex structures like the infrastructure assets [Makkonen, 2016]. Semantic 3D models improve the data access and create more transparent workflows between different sectors of public infrastructure management. Due to their high level of detail, the 3D models can be easily interpreted by anyone, which in turns increases the participation of citizens as asset users. Better cooperation between the stakeholders in AM facilitates the information sharing between different parties and improves the quality of

the projects [Altmaier and Kolbe, 2003].

BIM is a well-known design method in the building and construction industry, which represents an information-rich and model-centric process that enables architects, engineers, and planners to create multidimensional models of various urban objects. Due to its highly detailed and precise modelling techniques, BIM can increase the clarity of project intentions for all stakeholders and ensure data fidelity and continuity across the asset utilisation period [Autodesk, 2012]. Therefore, BIM models are the most suitable sources of information for representing infrastructure assets and improving their life cycle.

This research focuses on the integration of 3D BIM models of civil structures in a Web GIS for public space management in order to improve the decision making in LCAM. OBSURV is a web application developed by Sweco Netherlands and used by multiple Dutch municipalities, provinces, and water authorities for a more efficient management of their assets. Currently, only 2D data visualization is available, which limits the functionalities of the GIS system to manage the entire asset life cycle in a more efficient way.

The 3D visualization of bridges is chosen for this project, as bridges are one of the most complex civil constructions and require a highly integrative approach. Obtaining reliable and timely assessments of bridge condition, performance and safety represents a very challenging task for bridge owners and engineers. Thus, it is essential to evaluate existing bridges in order to enable more cost-effective and efficient maintenance management decisions [Catbas et al., 2013].

1.2 RESEARCH OBJECTIVE

The goal of this research is to investigate the possibility to integrate 3D BIM models of civil structures in a Web GIS for optimising dynamic and complex processes in LCAM. Thereby, the challenges in collecting, processing, storing, and visualizing 3D spatial data will be presented and potential solutions will be provided. As a result, a 3D prototype will be developed by using WebGL, a well-known technology for web rendering of interactive 2D and 3D graphics and animations without the need of additional plug-ins. The prototype attempts to demonstrate a possible solution for the integration of 3D models in a GIS system for AM. The solution should facilitate a better management and representation of infrastructure assets in the maintenance phase. The project is made in cooperation with the Municipality of Rotterdam due to their strong interest in the 3D visualization of infrastructure assets in OBSURV.

The research covers a broad scope including the collection, manipulation, storage and representation of semantically rich 3D spatial data. The main objective is to investigate the current data visualization in LCAM, collect and process the 3D data, and find the most suitable WebGL technique for the visualization of complex 3D models. The project aims to answer the following research questions:

1. *What is the current state of data visualization in LCAM and how can it be improved?*

   Initially, it is necessary to understand the meaning of LCAM and its processes, and investigate the current practices of visualizing public infrastructure assets in order

to determine the potential improvements. Furthermore, it is essential to research the added value of semantic 3D models within the LCAM concept.

2. *What are the requirements for the development of the 3D prototype?*

   With respect to the integration of 3D models in a Web GIS, it is crucial to determine the stakeholders in the project and gather their requirements for the main functionalities of the 3D prototype.

3. *How can 3D BIM models be processed for effective utilisation in condition monitoring?*

   The BIM models should be processed accordingly in order to be used for the condition monitoring of assets as part of the maintenance and operation LCAM phase. For that purpose, it is essential to familiarise with the national standards, which specify the normalised composition of the public infrastructure assets.

4. *How can 3D BIM models be integrated in a Web GIS?*

   The integration of 3D data is the major component in the research and requires special attention. The 3D prototype should be designed and developed according to the requirements given by the stakeholders.

The research follows several steps to achieve the desired goal. First, it is essential to get acquainted with the LCAM concept, Web GIS, and the semantic 3D models by reviewing the scientific literature. Then, the stakeholders in the project are identified and their requirements are gathered in order to create an appropriate design concept of the prototype user interface (UI). After that, the prototype is developed by using WebGL. In addition, some implementation issues are presented and strategies to solve them are provided. At the end, the prototype is evaluated to ensure that the user preferences are met to a large extent and allow the users to give any improvement suggestions.

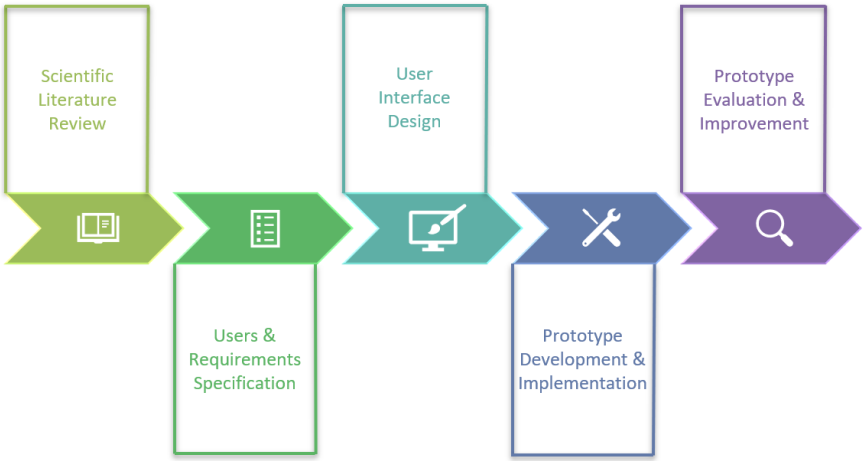


Figure 1.1: Process of the prototype development.

1.3 THESIS OUTLINE

After describing the research objective and research questions, a brief overview of the content of the thesis is presented. The document is structured as follows.

Chapter 2 contains background information about LCAM and its phases, Web GIS along with its characteristics and standards, and semantic 3D models. It is important to understand the meaning of the terms used in the research in order to proceed with the 3D integration. In addition, the chapter gives an overview of the related work on the deployment of 3D models in different application areas and the georeferencing of BIM and other models.

Chapter 3 focuses on the design of the 3D prototype taking into account the functional and non-functional requirements given by the stakeholders in the project. The UI of the application is presented in the form of a mockup model, which combines all the desired elements of the 3D prototype. Based upon this, several WebGL frameworks are described aiming to find the most suitable technique to develop the 3D prototype according to the user preferences.

Chapter 4 presents the sample dataset to get familiar with the provided BIM model and the design platform used for its modelling. Then, the technique for the decomposition of the 3D model is explained as part of the data processing.

Chapter 5 is the main part of the thesis, as it contains the development of the 3D prototype. First, the preparation of the test environment is addressed. Consequently, the configuration of the 3D viewer in WebGL is presented along with its implementation and some issues related to its operation.

Chapter 6 includes the prototype evaluation done by different stakeholders in the AM and BIM fields in order to judge the application usability and propose any possible improvements. Subsequently, the proposed approach for the integration of BIM models in Web GIS systems for public space management is critically assessed and discussed.

Chapter 7 concludes the thesis by answering the research questions and providing suggestions for expanding the research in the future. The improvement recommendations are given partly based on the feedback from the prototype evaluation. In addition, the contribution of the research to the Geomatics field is discussed.

Appendix A illustrates the structure of the initial BIM model and Appendix B the structure of the decomposed BIM model, as specified in Autodesk Inventor, the BIM modelling platform used in this research.

Appendix C presents a diagram of the tables stored in the OBSURV database and the relations between them.

Appendix D contains the HTML, CSS, and JavaScript codes for the configuration of the 3D prototype, and Appendix E the PL/SQL code for the procedure for retrieving data from the OBSURV database.

Appendix F includes the questionnaire survey for the evaluation of the developed prototype. The questionnaire is extracted from Google Forms.

Appendix G encompasses the thesis reflection.

2 THEORETICAL BACKGROUND

After the main objective of the research has been described, theoretical background on the studied topic is provided to extend the knowledge and understanding of the phenomena. First, the LCAM concept is explained along with the advantages of integrating 3D models in the work procedures. Then, the characteristics of Web GIS systems are described and the test application for the research, OBSURV, is introduced along with its services and architecture. Consequently, the most common types of semantic 3D models are presented and related studies on the web visualization of 3D models are reviewed.

2.1 LIFE CYCLE ASSET MANAGEMENT

LCAM is an integrated approach for optimising the life cycle of infrastructure assets in order to improve the decision-making and scheduling processes, and enhance the product management and organisational structure in AM [Wolter, 2010]. The framework aims to determine the overall life cycle cost and value of an asset by collecting information about the asset from its planning phase to its demolition [NAMS and IPWEA, 2011]. Too [2008] clusters the core stages in LCAM into asset planning, asset creation, and asset maintenance and operation. Each life cycle phase includes several different activities (see Figure 2.1).

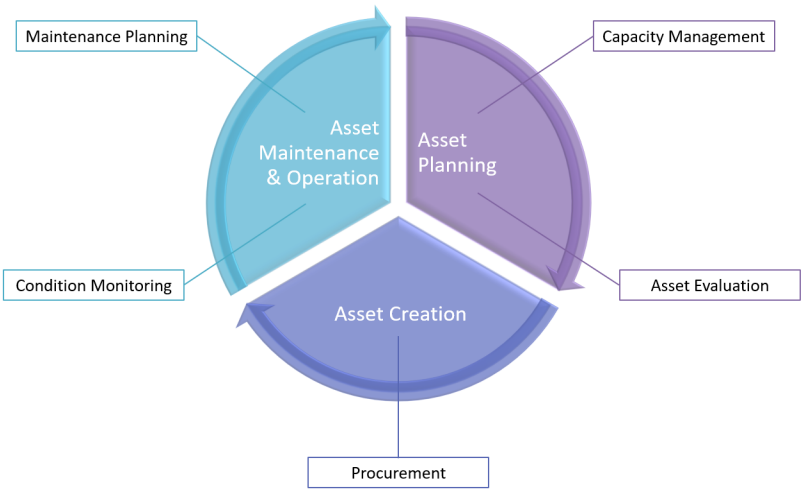


Figure 2.1: Life cycle of infrastructure assets, adapted from Too [2008].

Asset planning encompasses corporate strategies created by asset managers to determine the operation goals and needs by focusing on capacity management and asset evaluation. Capacity management is necessary for estimating the ability of infrastructure assets to meet future requirements in order to prevent capacity failure or underutilisation. In the next stage, the assets must be evaluated to improve their life cycle and

maximise their economic, social and environmental benefits [Too, 2008].

Asset creation involves the acquisition, construction and commissioning of new infrastructure assets. In this life cycle phase, a good project management is crucial for the project organisation, which includes financing, tendering, procurement of all the associated materials, organisation and obtaining the required human resources. Furthermore, the efficient project management can reduce potential risks related to the planning, construction, commissioning and operation of the assets [Too, 2008].

Asset maintenance and operation are key components within the LCAM concept. The maintenance process aims to improve the operational efficiency and effectiveness of existing infrastructure assets in order to maintain a high service capacity. The maintenance management includes maintenance planning and condition monitoring. Maintenance planning is recognised at all levels of the industry due to the increasing demand pressure on infrastructure assets. Condition monitoring provides important information about the current condition and capacity of the assets, and is essential for their optimisation [Too, 2008]. NEN 2767-4 are Dutch national standards for determining the condition of the public infrastructure in an objective and unambiguous way through visual inspections and physical checks of the assets. The norms are created to ensure a proper management of the assets by providing insight into their current state and maintenance situation, and facilitating maintenance planning, budgeting, and prioritisation [NEN, 2018b].

NEN 2767-4 enables to determine the condition of various infrastructure assets, such as roads and rail roads, nature and green areas, sports fields and playgrounds, bridges, tunnels, ports, embankments, sewage systems, cables and pipelines, etc [NEN, 2018a]. According to the standards, the assets are represented on three separate levels: objects, object elements, and object components (see Figure 2.2). The condition score specified for each object component ranges from excellent to very bad [Boanca, 2014]. In addition, any possible defects of the components that require reparation are reported in the inspection documents. The NEN standards are implemented by the majority of Dutch municipalities and other public institutions. Thus, the infrastructure assets in OBSURV are stored in compliance with the levels specified in the NEN 2767-4 standards.

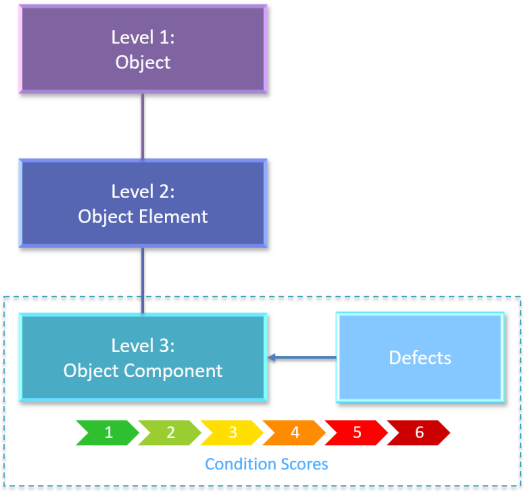


Figure 2.2: NEN levels for the infrastructure assets, adapted from Boanca [2014].

In the last two decades, there has been an increasing growth in the GIS development due to the necessity for acquisition, storage, manipulation, and visualization of spatial data. GIS is primarily used for mapping, but also for performing complex spatial analysis and creating 3D models of the built environment [Scianna, 2013]. The GIS systems contain essential information about the location of geographic features and their descriptive attributes, which can be used for spatial modelling and analysis in order to deliver new insights [Bodzin and Anastasio, 2006]. Recently, the capabilities of GIS have expanded to complex 3D visualization due to the increasing need for 3D geoinformation in urban planning, cadastre, environmental monitoring, landscape planning, and infrastructure and utility management [Zlatanova et al., 2002].

For all these reasons, GIS has significantly improved the operation in LCAM by enabling advanced data modelling, management, and visualization. Previously, the systems in LCAM used simpler administrative tools like tables, which limited the data analysis and visualization opportunities. The current systems are designed to produce reports and thematic maps for the needs of the asset managers, and enable them to add and delete objects and update their semantic properties. Traditionally, 3D models are incorporated in the planning and construction LCAM phases for design purposes [Makkonen, 2016]. However, the asset maintenance and operation are still based on 2D data and reports from visual inspections. Therefore, 3D data can be integrated in this LCAM stage to ensure a better future management of the infrastructure and usage of the information provided in the asset design and planning for its maintenance. In this case, the advances in the 3D GIS systems come into play to ease the integration process.

3D modelling is used to connect the reality with design models, which can be beneficial for various aspects of LCAM projects focused on capturing of existing physical conditions. In that sense, the 3D models enable a more realistic and detailed representation of the built environment and underground space [Autodesk, 2012]. 3D models are very powerful tools for the visualization of subsurface assets like pipelines and cables, as they can clearly show the location and direction of the parallel assets, which is difficult to represent in a 2D environment [Mendez et al., 2008].

Due to their ability to represent the real world, 3D models create more transparent and automated work processes between different sectors of public space management [Altmaier and Kolbe, 2003]. Therefore, the 3D data can be easily accessed and analysed by experts and non-experts at different LCAM stages. The cooperation between multidisciplinary stakeholders leads to better decision making and reduced risk [Fredericque and Lapierre, 2010].

The possible long-term use and re-use of detailed 3D data helps to avoid repeated work and improve the project efficiency. The interoperability and compatibility of the 3D models offer more convenient data updating, which assures the sustainability and quality of 3D data resources [Altmaier and Kolbe, 2003].

Despite all the evident advantages of 3D models, their implementation requires significant changes in the existing LCAM systems related to the construction, management, and integration of 3D data. First, more powerful hardware and software are necessary for 3D modelling and rendering, as 3D models are heavier than 2D data. In addition, the 3D data need to be stored efficiently to allow for data interoperability and database

integration [Makkonen, 2016]. Furthermore, considerable amount of time is needed for training the staff to deploy new technologies in their work processes and follow new methods for the management of their assets [Hixon, 2012].

2.2 WEB GIS

The Geographic Information Infrastructure (GII) is a framework, which facilitates the availability and access to geoinformation for governmental institutions, private organisations, non-profit sector, academia, and citizens [van Loenen, 2009]. The infrastructures are composed of multilayered spatial data, database management system (DBMS) and processing services to store and retrieve data, and interoperability standards to enable the integration of the different datasets [van Oosterom et al., 2000].

Due to recent developments in the web technology, traditional GIS has transitioned to Web GIS, which improves the availability and dissemination of geoinformation to a great extent. Web GIS has evolved from the delivery of static maps provided in raster formats to dynamic maps generated directly from spatial databases. In that sense, the web applications enable interactive data representation allowing the users to view, query, analyse, and freely download georeferenced data [Paiva and Baptista, 2009].

The web distribution of GIS is enabled through a client-server architecture to allow easier access to the database. The communication between the client and server is established by using HyperText Transfer Protocol (HTTP), i.e. the client sends an HTTP request to the server and the server returns an HTTP response with the requested data to the client [Mozilla, 2018c]. The server retrieves the spatial data directly from the database on the server side (see Figure 2.3).

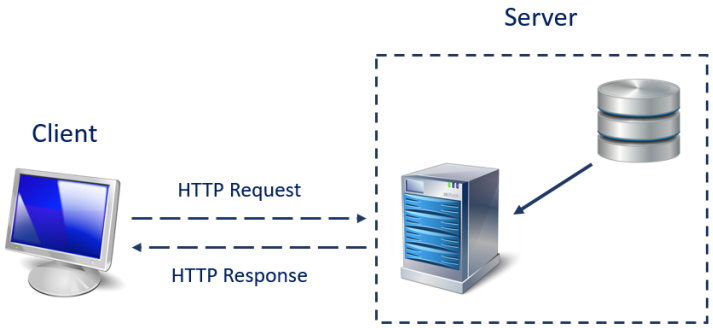


Figure 2.3: Client-server architecture, adapted from Alesheikh et al. [2002].

In order to improve spatial data exchange and interoperability, the Open Geospatial Consortium (OGC)[1] defines various standards for data models and web services adopted in the GIS community [Sayar and Pierce, 2005]. The most commonly used OGC web services for web mapping are the Web Map Service (WMS), Web Feature Service (WFS), and Web Coverage Service (WCS), presented in Figure 2.4.

1 International non-profit organisation, leading the development of open standards.

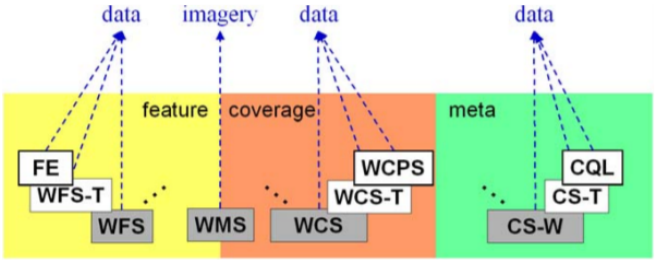


Figure 2.4: OGC standards for web mapping [Baumann and Aiordachioaie, 2009].

WMS produces maps of georeferenced data as digital image files suitable for display on a computer screen. The generated maps are rendered in a pictorial format such as JPEG, PNG, or as vector-based graphical elements in Scalable Vector Graphics (SVG) format. The service provides several operations for returning maps with well-defined geographic and dimensional parameters, as well as information about particular features shown on a map, and service-level metadata. The client requests can be submitted directly by using an Uniform Resource Locator (URL) [de la Beaujardiere, 2006].

WFS provides direct access to geoinformation at the feature and feature property level. It stores and serves the spatial data encoded in Geography Markup Language (GML), which allows the users to query features based on spatial and non-spatial constraints. Furthermore, the service provides transaction operations for creating, updating, or deleting feature instances, and a process for locking the features during a transaction [Sayar and Pierce, 2005; Vretanos, 2005].

WCS offers access to multi-dimensional coverage data to support the networked spatial data exchange. The service provides available data along with their detailed descriptions and returns data with its original semantics, which can be used for data interpretation and analysis. WCS allows the client to request a coverage comprised of selected range properties at a selected set of spatio-temporal locations [Whiteside and Evans, 2007].

OBSURV is a Web GIS for inventory making, inspection planning, budgeting, and managing of public infrastructure assets (see `http://www.obsurv.nl`). It has a simple and user-friendly interface designed to serve both technical and non-technical customers. The main users of the application are the asset owners, managers, and executors working in the Dutch public sector.

OBSURV is available on-premise and on hosting and operates on multiple devices, such as computers, laptops, tablets and smart phones. It offers several modules for public space management, such as road infrastructure, sewage systems, green areas, public lighting, and civil constructions. This allows the users to choose the desired module(s) and use the provided tools for the efficient management of their assets.

The majority of data are represented in a tabular form in order to prevent disruption to the existing working practices of the administration. In addition, a geographic map is provided in each module to enable the users to view, add, modify, or delete their data for keeping the information accurate and up-to-date. The application complies with the OGC standards for web mapping by using GeoServer[2] to share and edit geospatial

2 Open source server designed to publish data from various spatial data sources.

data and working with web services, such as the WMS for providing georeferenced map images and the WFS for querying geographical features [Open Source Geospatial Foundation, 2017].

OBSURV has a multi-tier architecture, which consists of a database server for data storage and web server for data transmission on the Internet. The database server is connected to the web server for providing services to the web browsers (see Figure 2.5). In order to develop the 3D prototype, it is necessary to better understand the functionality of each component in the software architecture.

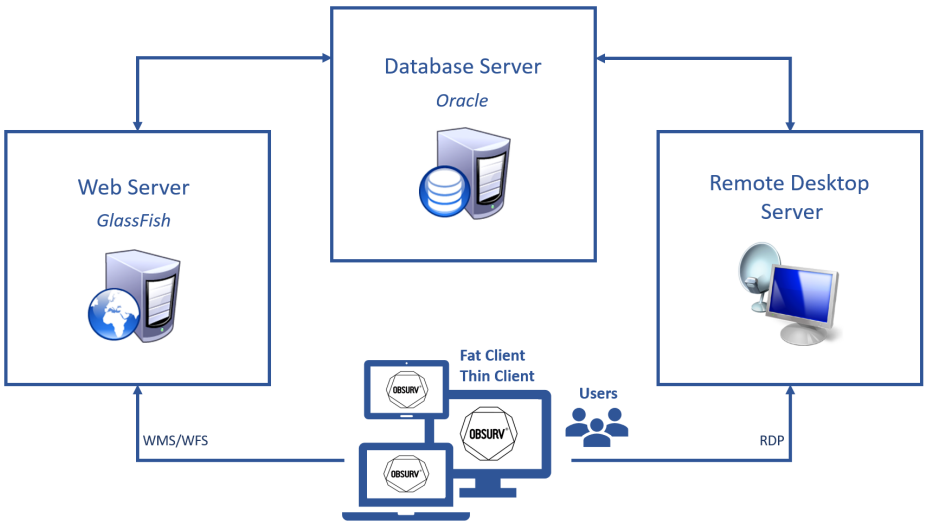


Figure 2.5: Multi-tier client-server architecture of OBSURV, adapted from Tepas [2014].

OBSURV utilises several external applications, such as the Publieke Dienstverlening Op de Kaart (PDOK)[3] viewer, GlobeSpotter[4] street view images, and BGT[5] layers for enhancing the data visualization. Oracle is highly integrated in the front-end and back-end application development. The Web GIS is designed by using Oracle Application Express (APEX) due to its ability to create responsive, database-driven applications with low level of coding [Oracle, 2018]. OBSURV is deployed on the GlassFish[6] server. In order to ensure full operability of the application, the server hosts additional packages like the APEX Listener for establishing user process connection to the database and Jasper Reports Integration package for generating reports and exporting them as PDF, Excel, Word, etc.

Spatial datasets are usually very large and require an efficient method for data storage and management. The database server hosts Oracle Locator, a spatial extension of Oracle 11g Standard Edition, where the public assets data are stored (see Figure 2.5). The spatial database can be manipulated by using Structured Query Language (SQL) and its procedural extension Procedural Language/Structured Query Language (PL/SQL) as the key programming languages embedded in Oracle. Each application module has a well structured schema with a great variety of tables and combined views to facilitate better database management.

3 Dutch central facility, which provides governmental geospatial data.
4 Web-based application which provides cloud access to imagery created by CycloMedia.
5 Basic registration large-scale topographic map of geographic elements like roads, water, buildings, etc.
6 Java open source web server sponsored by Oracle.

The hosting version of OBSURV runs on a remote desktop, which allows the users to connect to another computer over a network connection by using the Remote Desktop Protocol (RDP). In fact, the majority of OBSURV users deploy the hosting version of web application instead of the on-premise version. The remote desktop server is directly linked to the database server for retrieving the spatial data (see Figure 2.5).

2.3 SEMANTIC 3D MODELS

This section introduces the most common types of semantic 3D models, which are BIM and CityGML, in order to give a better overview of the sources of 3D geoinformation that are discussed throughout the thesis.

2.3.1 BIM

BIM is a combination of organisational solutions and technologies for improving the quality of the design, construction, and maintenance of buildings and the built environment (see Figure 2.7). The main objective of the BIM process is to increase the productivity and efficiency in the construction industry by eliminating design errors, helping the management of processes in construction, and deepening the collaboration between different stakeholders [Miettinen and Paavola, 2014]. Therefore, BIM is considered to be the most promising approach in the Architecture, Engineering and Construction (AEC) industries [Eastman et al., 2011]. In that regard, BIM facilitates the generation, usage, support and maintenance of specific products and services, such as 3D models and data related to the life cycle of an asset [Succar et al., 2013].

The BIM technology generates computer-based models, which contain all physical and functional characteristics of a built facility through its life cycle. The BIM models can be created in many dimensions based on the desired level of development. The simplest models are in 2D to represent the horizontal X and vertical Y dimensions. These are usually extended to 3D design models, which include the depth Z as well. In most cases, the BIM models combine 4D for construction duration information and sequence with 5 for materials costs [Alexiadi and Potsiou, 2012].

Initially, the level of detail required for each design stage in BIM was not clearly defined to the construction engineers. In order to add clarity to the building process, the American Institute of Architects (AIA) and BIMForum introduced the level of development (LOD) concept, which specifies the geometry and attached information that should be represented at the various levels [Yoders, 2013]. BIMForum [2016] distinguishes five levels, in which the BIM models become more detailed and informative with increasing LOD:

- **LOD 100:** The model element is geometrically represented. Additional information can be attached to show the existence of a component but not its shape, size, or precise location.
- **LOD 200:** The model element is graphically represented as a generic system or object and it can be recognised as the components it represents. Additional information can be attached to specify the quantities, size, shape, location, and orientation of the element.

- **LOD 300:** The model element is graphically represented as a generic system or object. Additional information can be attached, but the quantities, size, shape, location, and orientation of the element can be measured directly from the model.
- **LOD 350:** The model element is graphically represented as a generic system or object and parts necessary for coordinating it with nearby elements are modelled. Additional information can be attached, but the quantities, size, shape, location, and orientation of the element can be measured directly from the model.
- **LOD 400:** The model element is graphically represented at sufficient detail and accuracy for fabrication of the represented component. Additional information can be attached, but the quantities, size, shape, location, and orientation of the element can be measured directly from the model.

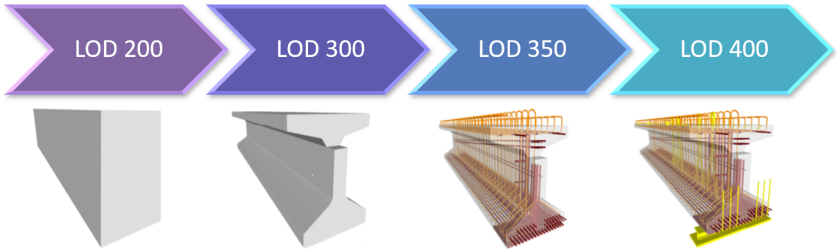


Figure 2.6: LOD in the BIM technology specified by BIMForum [2016].

The BIM models are designed by using object-based parametric modelling, which is the major difference to traditional Computer Aided Design (CAD) systems. The approach uses predefined parameters and rules to represent the geometric and semantic properties of a built facility. In that sense, BIM allows for the automatic adjustment of the object geometry to any changes made by the user, whereas CAD requires a manual editing of each geometry aspect [Eastman et al., 2011].

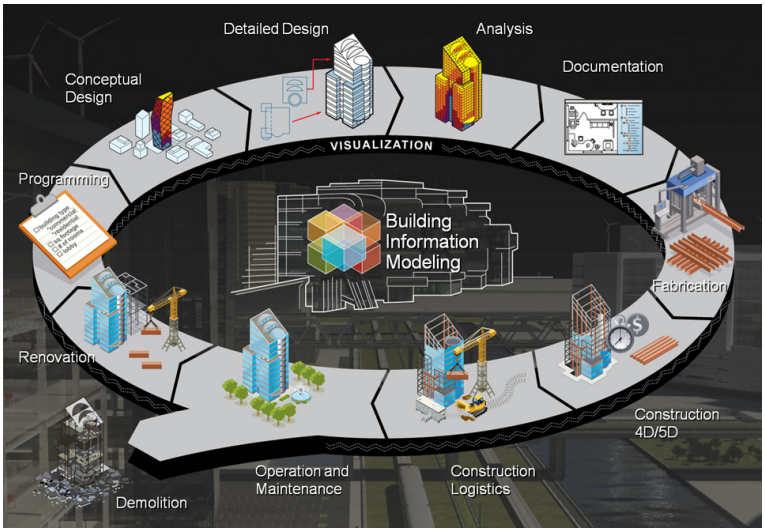


Figure 2.7: BIM implementation in the building life cycle [Dispenza, 2010].

Figure 2.7 shows the implementation of BIM processes and technologies during all phases of an asset life cycle. BIM models bring considerable benefits to the planning, design, construction and operation of the built assets. The building performance and quality can be increased by the development of a schematic model prior to the generation of a detailed model to validate the correctness of the proposed scheme and

determine whether it meets the specified requirements. Furthermore, BIM provides enhanced visualization of complex geometries by implementing parametric modelling as well as efficient data storage, information sharing and exchange [Vanlande et al., 2008; Eastman et al., 2011]. This considerably improves the quality of work and coordination among the design and engineering processes [Alexiadi and Potsiou, 2012]. For the construction logistics, the detailed BIM models can be linked to existing construction plans for simulating the construction process and identifying any potential problems related to it. The information contained in the models can be used for monitoring the condition of the built facility after its construction and controlling its operation and maintenance [Eastman et al., 2011]. In that sense, BIM enables a better estimation of future needs in the construction industry and contributes to an extended asset life [Makkonen, 2016].

Despite the benefits that BIM modelling brings to the AEC industry, the BIM adoption is relatively slow due to technical and managerial obstacles. The technical complexities are related to the data interoperability and computability, and integration of information among the BIM model components. These problems can be overcome by using well-defined construction process models and developing practical strategies [Alexiadi and Potsiou, 2012]. Although BIM offers new methods for collaboration, it can also cause organisational issues concerning its implementation. The biggest problem in the BIM development is the use of different modelling tools within a team and the necessity to migrate from one environment to another. This is a time-consuming and error-prone process, which can be reduced by using standards for data exchange [Eastman et al., 2011].

There is a wide range of commercial and open-source BIM software suitable for the planning, design and construction of built facilities [Makkonen, 2016]. The BIM platforms support diverse file formats, such as Industry Foundation Classes (IFC), Standard for the Exchange of Product Data (STEP), Initial Graphics Exchange Specification (IGES), Standard Tessellation Language (STL), Wavefront OBJ as well as design and drawing formats like DNG, DWG, DWF, and DXF. The leading BIM technology providers are Autodesk, Bentley, and Tekla.

Autodesk is the best-known software provider for 3D design and engineering. The most common products are Autodesk Revit for conceptual design, Autodesk Civil 3D for civil engineering and construction documentation, Autodesk Inventor for mechanical design, and Autodesk Maya for 3D animation, modelling and simulation [Autodesk, 2018]. The BIM software used in this project is Autodesk Inventor (see Section 4.1).

Bentley offers a variety of products for architecture, engineering, and construction. Its integrated systems include Bentley Architecture for architectural modelling, Bentley Structural for structural design, Bentley Building Mechanical Systems for mechanical engineering, Bentley Facilities for facility management, and Bentley PowerCivil for site planning [Eastman et al., 2011].

Tekla focuses on the development of BIM software for structural design and engineering. Its main products are Tekla Structures for structure modelling, Tekla BIMsight for construction project collaboration, and Tekla Civil for managing infrastructure data [Tekla, 2018].

2.3.2 CityGML

Currently, the use of virtual 3D city has gained significance in many application fields like urban planning, disaster management, facility management and environmental simulations. The complexity of these real-world use cases requires the spatial and geometric representation of the 3D objects as well as their thematic properties and logical relationships [Kolbe, 2009]. In this light, CityGML is very efficient for creating 3D city models, as it combines the graphical and semantic information about urban objects [Mao, 2011]. The virtual 3D city models of Berlin, Potsdam, Rotterdam, and Helsinki are among the most popular models created with CityGML data [CityGML Database, 2018].

CityGML is an OGC standard for the storage and exchange of 3D geoinformation. It is based on GML3, which inherits the GML geometries defined as points, curves, surfaces, solids, and composition objects [Zlatanova et al., 2012]. CityGML is functionally partitioned into modules to allow for partial CityGML implementations. The CityGML core, generics, and appearance define structures that are applicable to a variety of urban objects, such as buildings, bridges, tunnels, traffic infrastructure, water bodies, vegetation, and city furniture [Kolbe, 2009]. The standard defines the relations for the aforementioned objects with respect to their geometrical, topological, semantic, and textural properties [Gröger et al., 2012].

CityGML uses a multi-scale level of detail (LOD) concept with increasing accuracy and structural complexity to facilitate efficient spatial data analysis and visualization by enabling independent data collection process dependent on the model requirements [Gröger et al., 2012]. The concept differs substantially from the LOD concept specified for the BIM technology. Kolbe [2009] defines the following five consecutive levels:

- **LOD0:** A Digital Terrain Model (DTM) is represented in 2.5D.
- **LOD1:** A blocks model is represented without including any roof structures.
- **LOD2:** A building with distinctive roof structures and larger building installations is represented.
- **LOD3:** Architectural model with detailed wall and roof structures, doors and windows is represented.
- **LOD4:** Architectural model is represented, including interior structures, such as rooms, stairs, and furniture.

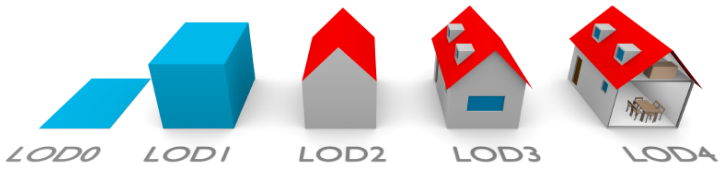


Figure 2.8: LOD in CityGML [Biljecki et al., 2017].

These levels aim to represent the accuracy of the objects, i.e. LOD0 has the lowest resolution (meters) and LOD4 has the highest resolution (millimetres). This makes the difference between CityGML and the traditional computer graphics techniques, which are intended for fast data visualization [Zlatanova et al., 2012].

The coherent semantic-geometric modelling is fundamental in the CityGML design, as the semantic information can reduce the ambiguities for geometric integration, if it is structured with respect to geometry. The semantic model consists of class definitions for the urban objects, associated with attributes, relations and aggregation hierarchies. The geometric properties determine the geographic location and extent of the features. The GML geometries can be combined for each dimension to form aggregate or composite geometries (see Figure 2.9). In CityGML, the topology of the features can be represented explicitly by modelling each part once and referencing by all features, which include the same geometry [Stadler and Kolbe, 2007].

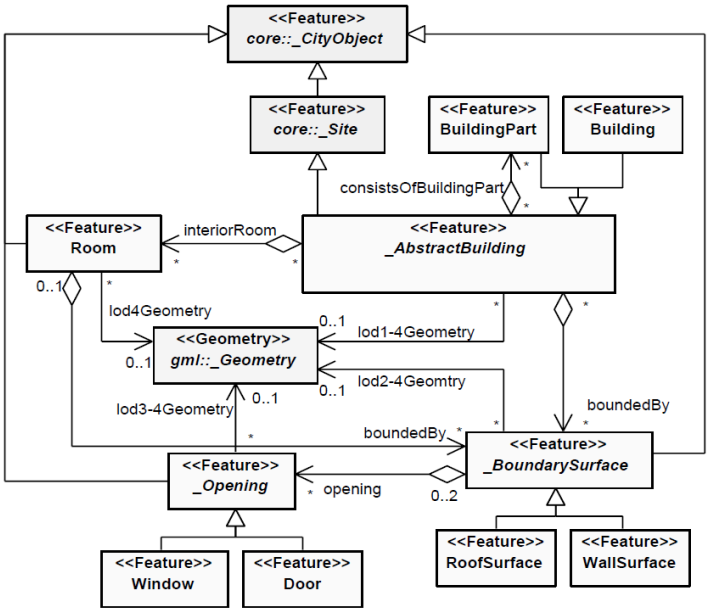


Figure 2.9: UML class diagram of the semantic and geometric properties for the building model [Kolbe, 2009].

In addition to the spatial properties and semantic information, CityGML defines appearance for the features, which represents themes like infrared radiation, noise pollution, or earthquake-induced structural stress for a more realistic representation. The appearance is composed of data for each surface geometry object. In that case, the geometry objects can have surface data for multiple themes and surface data can be shared by multiple geometry objects [Gröger et al., 2012].

CityGML exists as GML file as well as spatial schema for relational DBMSs. Therefore, the CityGML models can be imported to 3D City Database (3DCityDB), an open source spatial database containing a set of tools for the efficient storage and management of large 3D city models. The database schema can be implemented on Oracle Spatial and PostGIS by using the geometry data types specified for each DBMS. Furthermore, the database allows to export the geometric properties of the city models in different formats suitable for visualization, such as COLLADA [Prandi et al., 2015; Zlatanova et al., 2012].

2.4 RELATED WORK

In recent years, the integration of 3D models in Web GIS has been extensively researched in diverse fields. The majority of studies focus on the creation of 3D city

models by using CityGML data and their visualization in WebGL. The solutions presented below provide an overview of several approaches for the web visualization of CityGML models in Cesium, a WebGL framework for representing dynamic spatial data that supports various 2D and 3D geometries [Chaturvedi, 2014].

Schilling et al. [2016] presented a method for the real-time streaming of large and highly detailed CityGML models based on open standards. The researchers used GL Transmission Format (glTF) for encoding the 3D models and 3D Tiles for associating the glTF objects with their attributes and streaming the 3D data. The Batched 3D Model (B3DM) is implemented within the 3D Tiles concept for transferring multiple 3D models in a single request and rendering them efficiently in WebGL by using batches [Cozzi et al., 2018]. Various 3D models of buildings and urban objects were visualized in Cesium to improve the data exchange and interoperability (see Figure 2.10).



Figure 2.10: 3D city model of Berlin visualised in Cesium by Schilling et al. [2016].

A similar research was conducted by Prandi et al. [2015] who developed an approach for accessing large CityGML data through web services by separating the geometric and semantic parts of the 3D models. The researchers used NASA World Wind (NWW) and Cesium for the visualization of the geometries. Both web technologies are open source virtual globes and allow for an efficient representation of 3D city models. NWW utilises a Java applet with an optimised custom-rendering engine for 3D buildings (see Figure 4.8a). The CityGML data are loaded in a custom servlet and transformed into the C3D file format, readable in NWW. The other option is to integrate CityGML models in Cesium by using C3D and glTF, which was considered by the authors as a more promising solution (see Figure 4.8b). The semantic information about the 3D models can be retrieved from the 3DCityDB database by using the WFS service.



(a) (b)

Figure 2.11: 3D city models of Rotterdam created by Prandi et al. [2015] in: (a) NWW applet. (b) Cesium.

Chaturvedi [2014] developed a web-based application for visualizing virtual 3D city models and performing 3D buffer analysis by using WebGL. In his research, the author built a decision support system for emergency situations that can be utilised by decision makers and field workers for evacuation planning. The CityGML data were exported to Keyhole Markup Language (KML) format, which is readable for any JavaScript libraries. The virtual globe in Cesium was used for rendering the 3D models. Several algorithms were computed to generate 3D buffer zones around the affected buildings on-the-fly and determine which urban objects are located within these buffer zones (see Figure 2.12).

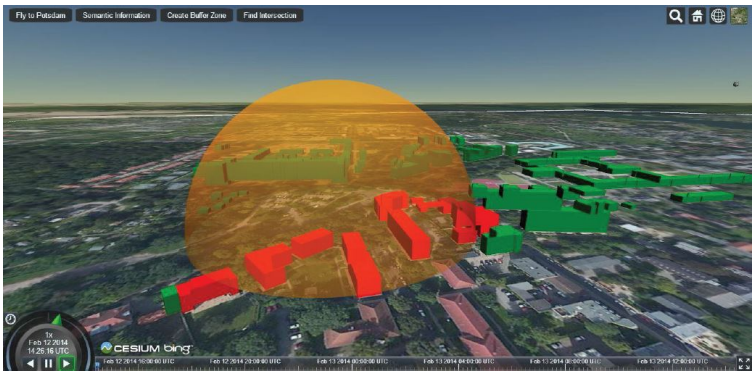


Figure 2.12: Web platform for 3D buffer analysis developed by Chaturvedi [2014].

The presented studies focus mainly on the creation and web-based visualization of 3D models of buildings. However, there has been very limited research on the use of semantic 3D models of infrastructure assets in GIS, which can be integrated in public space management systems for optimising the decision-making processes.

Makkonen [2016] conducted a research on the current practices for creating and maintaining semantic 3D models and the benefits of 3D geoinformation in AM. The author presented general requirements for an information system, which assures the efficient management of infrastructure assets, including reports generation, editable semantic and geometric properties of the objects, and ability to view and edit objects. Consequently, several commercial and open source software systems for semantic 3D modelling, which comply with the aforementioned requirements were reviewed. In addition, multiple experts in the AM field were interviewed to identify the practical benefits and challenges of semantic 3D implementation in public space management, and possible steps to overcome the challenges. The conclusion is that appropriate software solutions with proper data visualization are missing, although the need for integrating 3D models is increasing.

Boanca [2014] investigated the possibilities for integrating BIM models in OBSURV for a better public space management. In her thesis, she presented an approach for the simplification of a detailed BIM model through geometric transformation and semantic mapping in order to provide the required information for the maintenance phase. The simplified model is stored in the Autodesk 360 cloud and linked to the web environment via URL link to make the model accessible on any device (computer, tablet, mobile phone). The author provided a comprehensive insight into the generalisation of BIM models and presented an approach for the BIM-GIS integration. However, the functionalities of the created solution are limited only to visualization without allowing the users to see and edit the information about distinct elements in the model.

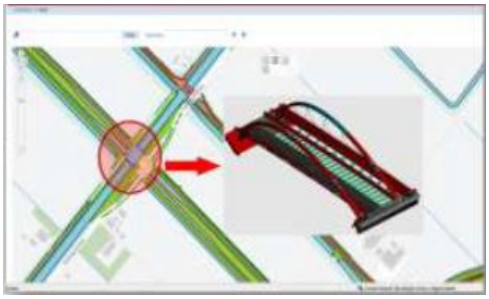


Figure 2.13: BIM model of a bridge integrated in OBSURV by Boanca [2014].

The integration of a base map and georeferencing of BIM models poses a major challenge in the BIM-GIS integration. Thus, several methods for defining the geolocation of diverse 3D models are reviewed to find an appropriate approach. Diakite [2018] researched the integration of geographic coordinates of a construction project in its BIM model in order to keep the information for a GIS application. The approach is based on 3D model of a building designed in Autodesk Revit and exported in the IFC format, which allows for georeferencing easier than other BIM standards. The geolocation of the BIM model can be determined by using the georeferencing tools in Revit (v2018). The real-world coordinates of building's address were defined in the provided mapping service to find the exact location of the asset on the map. Then, a project base point and a survey point represented in the floor plan of the building were used to rotate and position the 3D model correctly. The presented approach shows that the geolocation of some BIM models can be obtained in Revit, but the majority of BIM modelling platforms do not provide georeferencing tools. Furthermore, the proposed methodology cannot be applied to other BIM formats, as they do not have the same properties as IFC and require additional processing in order to be georeferenced.

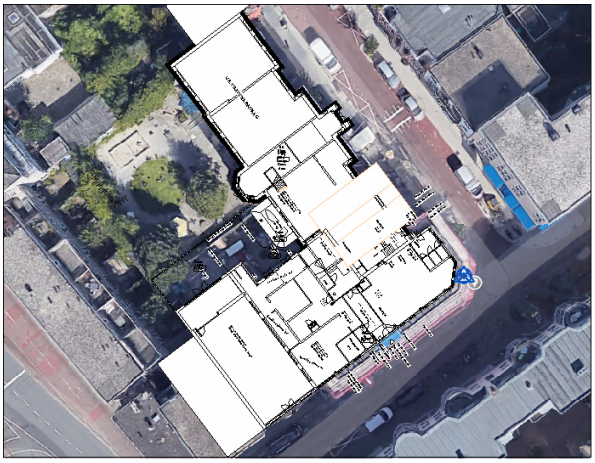


Figure 2.14: Georeferenced BIM model of a building in The Hague [Diakite, 2018].

Another research on the georeferencing of 3D models was conducted by Kolár and Wen [2009]. The authors presented a method for mapping and encapsulation of 3D objects to a geocentric coordinate system by associating any 3D model with a set of referencing parameters to find its location on the Earth. For that purpose, the geometries in the 3D model were normalised in a reference system with origin at the center of the model and a set of reference points on the map and in the model was chosen to aid the georeferencing procedure. The horizontal orientation of the points was specified to determine the angle between the south direction and the X axis of the model. Then,

the rotation and translation were applied to the normalised model to position it in correspondence with the geographic coordinates of one of the reference points. The proposed solution speeds the process of visualizing 3D models on the Earth surface by the automatic detection and transformation of vertices in the model.

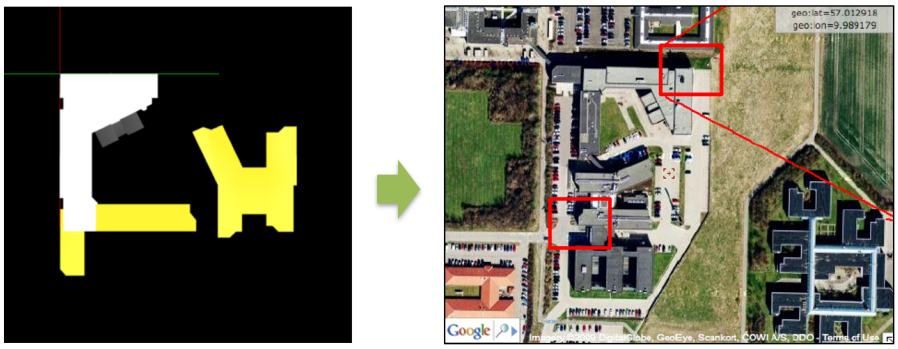


Figure 2.15: Georeferenced 3D model of a building by Kolár and Wen [2009].

Wang et al. [2013] developed a microscale meteorological model and integrated the Google Maps / Earth frameworks to improve the system functionality and usability. Additionally, the authors used high-resolution images, street views, and 3D buildings provided in the Google platforms to generate small-scale vegetation and building morphology data. The web mapping platforms use the Geodetic Coordinate Reference System (CRS) to represent locations on the Earth. In order to incorporate Google Maps / Earth in the meteorological model, it is necessary to convert the coordinates to a Cartesian CRS meant for analysis and visualization. However, the coordinate conversion results in a new CRS with an origin at the center of the Earth, which is not desirable for small-sized models, such as the meteorological model. Thus, a local reference system with an origin at the centroid of the model was constructed to reduce the distortion related to the curvature of the Earth's surface. The coordinate conversion and datum transformations are done by using well-known mathematical equations (see Section 5.2.3).

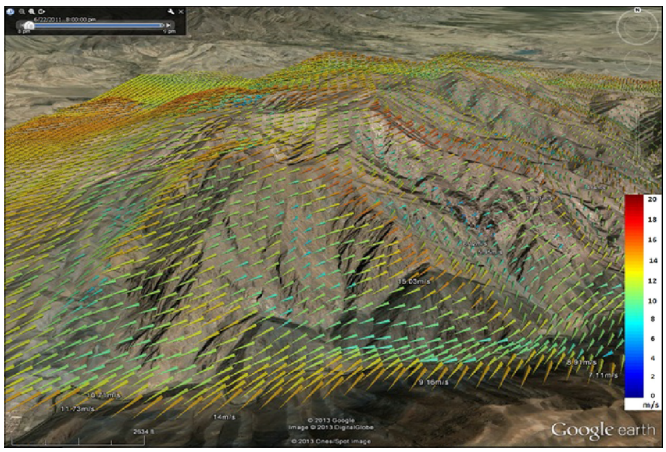


Figure 2.16: Meteorological model with integrated Google Earth platform created by Wang et al. [2013].

3 PROTOTYPE DESIGN

In the previous chapter the theoretical background was presented to get acquainted with LCAM, Web GIS, and semantic 3D models. This chapter focuses on the prototype design following the user centered design (UCD) approach, a multidisciplinary activity that incorporates users in the interface design to improve the quality of the end product [Voldan, 2012]. First, the stakeholders in the project are determined and their requirements are collected and analysed. Afterwards, a UI design concept is created according to these requirements. Subsequently, several WebGL frameworks are reviewed and compared to find the most suitable technology for the integration of 3D models in existing GIS web applications.

3.1 REQUIREMENTS SPECIFICATION

The UI design usually starts with a specification of requirements that come from the stakeholders to define the product or prototype objective and find a suitable approach for its design and development [Voldan, 2012]. The requirements describe the services that a system provides and the constraints on its operation [Sommerville, 2010].

The requirements can be gathered in different ways, such as direct observations by conducting on-site interviews, using surveys to collect information from many users, or focusing on existing case studies [Tidwell, 2010]. The integration of semantic 3D models in GIS systems for public space management is still a work in progress. Therefore, no standardised requirements for the development of a prototype, which visualizes 3D models of infrastructure assets, could be found in the scientific literature. The direct observations are considered as the best approach in the user research, as they provide precise information about the user goals and expectations [Voldan, 2012]. Therefore, the requirements for the development of the 3D prototype are specified based on interviews with potential users of the system.

The target group for the 3D prototype are AM experts, which coordinate the maintenance and operation of public infrastructure assets, such as asset managers in the municipalities and advisers and inspectors in consultancy companies like Sweco. Therefore, their opinions should be taken into account when specifying the requirements for the development of the 3D prototype.

Several interviews were conducted with different experts in the Municipality of Rotterdam, which are familiar with OBSURV, to better understand what can be expected from the 3D prototype and get more broad ideas about its implementation. In addition, specialists in Sweco were asked for their ideas and requirements related to the design and functional capabilities of the 3D prototype. The names and roles of the interviewees in this project are listed in Table 3.1 below.

Table 3.1: Interviewees participated in the graduation project.

Municipality of Rotterdam	Sweco Netherlands
Jan de Jong	Hilbert Davelaar
Project Leader OBSURV	*Product Manager OBSURV*
Joris Goos	Henri Veldhuis
Manager Digital Management & Building	*Manager Business Development*
Helmer Heijden	Marijn van den Berg
Asset Manager	*Asset Management Adviser*

Following the interviews with the municipality representatives, the results showed a variation of opinion and an overall lack of consensus with regard to the source of 3D geoinformation. Two out of three interviewees were in favour of CityGML data for the visualization of public assets, while one recommended the use of BIM models. On the other hand, the representatives from Sweco were unanimous in their preferences considering the same subject. They all agreed on the integration of BIM models in OBSURV due to their high level of detail and accuracy as well as the ability to manage the entire asset life cycle. CityGML is suitable for the visualization of multiple objects in a larger scale to give a better overview of the 3D environment, but the manipulation of the distinct objects is not so straightforward. Therefore, it is not certain whether a CityGML model of a bridge can be decomposed according to the NEN 2767-4 standards.

Due to the disparities in the view points of all stakeholders in this project, the development of the 3D prototype is in accordance with the requirements specified by Sweco and BIM models are chosen as data source. In order to ease the process of development and reduce potential risks in the project, the requirements should be prioritised according to their importance for the stakeholders. The requirements can be ranked on an ordinal scale from 1 being the most important to n being the least important requirement, with n being the total number of requirements [Business Analyst Learning, 2018]. The ordered list of requirements is presented below:

1. *Data collection, processing, and visualization*

   BIM models of civil structures must be acquired and processed in order to comply with the NEN 2767-4 standards for inspection and maintenance. The Municipality of Rotterdam is responsible for providing a 3D BIM model of a bridge that can be used for the integration.

   The processed 3D model must be visualized in a separate 3D web viewer integrated in OBSURV to make it easier for the users to distinguish between the 2D and 3D visualization. Overlaying the 3D data over the map is also avoided due to performance issues related to the large data volume. The BIM model must be linked to its corresponding 2D object stored in the database. Moreover, it is important to enable a subsurface view to represent the bridge elements, which are located below the ground level.

   Different colours must be assigned to the distinct bridge elements based on the inspection data to show their current condition. In addition, a legend must be

added in the 3D viewer to illustrate the meaning of the colours specified in the NEN 2767-4 inspections for a better understanding of the condition scores.

The bridge elements must be selectable and additional information about them must be shown in the 3D viewer. Furthermore, editable fields must be added to the UI to allow the user to update the information related to the asset materials, measurements, placement date, and documents.

2. *Base map and animation*

In addition, a base map must be incorporated in the 3D viewer to better visualize the surrounding environment of the bridge. In order to add a base layer to the 3D viewer, it is necessary to georeference the BIM model due to the lack of geolocation in the majority of BIM models. Thus, a suitable approach for georeferencing BIM models must be found.

The rotate, zoom and pan capabilities must be implemented for a better navigation of the view port and an enhanced visualization of the 3D model. An explode function can be added to the 3D viewer to separate the model into pieces and improve the visibility of the distinct parts.

3. *Additional functionalities*

A switcher can be incorporated in the 3D viewer to allow the users to control the visibility of the bridge elements. This functionality is useful, as it enhances the 3D model preview and makes the viewer more user-friendly.

In order to keep track of any changes in the condition of the bridge elements, it is desirable to include previous inspections and show the results in the 3D viewer by adjusting the colours of the BIM model.

As a result, a complete 3D prototype must be developed and the solution must be scalable to handle a greater amount of 3D data in the future. In order to achieve all the prerequisites, it is important to find a suitable technology for web rendering of interactive 3D models must be found, preferably open source, lightweight, and fast for smooth integration in existing web applications. WebGL is a good candidate for the development of the 3D prototype due to its compatibility with all modern web browsers. Furthermore, WebGL allows for customisation, which is important for the fulfilment of the aforementioned requirements.

The 3D prototype must be tested by several users or experts in the AM and BIM field to verify its operability, identify any errors made during the development phase, and determine whether it meets the predefined requirements.

3.2 USER INTERFACE DESIGN

UI design is a process of designing interfaces for computer and software applications with special focus on the layout and style. The design techniques aim to build user-friendly and easy-to-use interfaces, as users tend to assess the UIs based on usability and likeability. As the UI design plays a major role in the user experience (UX), a profound understanding of the user preferences is required to design satisfactory UIs

[Interaction Design Foundation, 2018].

The development of Virtual Reality (VR) systems in the last few years made the design of interfaces has made more complex. 3D UIs can be described as interfaces that involve 3D interaction, in which the users perform tasks directly in a virtual 3D environment. The 3D user interaction does not necessarily require 3D input devices like 3D mice to enable the communication between the user and the computer. The 3D interfaces can also be integrated in 2D applications [LaViola et al., 2017], as is the case with the 3D prototype. Nevertheless, the majority of studies on 3D UI design focus on the design of VR applications, which differs from the objective of this research.

The 3D interfaces can be very expressive to support more complex tasks, but they need to be designed in a simple manner to ensure user friendliness and full usability of each element of the interface [Soegaard and Dam, 2013]. Therefore, the prototype UI is kept simple in order to ease the navigation around the 3D viewer and increase the usability among both technical and non-technical users.

The graphical design of the 3D prototype is represented in the form of a mockup created with Moqups[1] (see Figure 3.1). Mockups reflect the design choices for the layouts of products and provide a more realistic perspective of the desired end result. When creating a mockup, it is important to consider the display of the content in terms of size and shape, and the visuals of navigation, such as a collapsible or drop down menu, forms, text fields, etc. Furthermore, the use of different colours and the colour contrast should be taken into account in the mockup design [Cao, 2016].

In OBSURV, the public infrastructure assets are represented in a tabular form and a collapsible page can be opened when the user selects an item in the table to display additional information about the object. The 3D viewer is integrated in this page so that the 3D model can be visible in a separate window (see Figure 3.1). The design of forward and backward navigation is essential for web applications and gives the users more freedom in their interaction with the system [Tidwell, 2010]. Thus, a close button should be incorporated in the collapsible page so that the user can leave the viewer and easily return back to the full view of the application.

For more efficient UI design, it is necessary to predict the first thing that the user expects to see in the application [Tidwell, 2010]. In this case, the 3D model should appear right after opening the collapsible page, as this is the central feature of the prototype. A base map should be integrated in the viewer to illustrate the surrounding area of the bridge either in 2D or 3D. A 2D base map can be added by using web mapping tools like Leaflet[2], Mapbox[3], and CartoDB[4]. Another option for the integration of a base map is to attach a plane to the BIM model and add an image of the surrounding area as texture of the plane. For the 3D visualization of the built environment around the bridge, it is necessary to incorporate an existing web viewer that renders the 3D objects in the defined area and gives a more realistic overview.

1 Web application for creating real-time wireframes, mockups, diagrams, and prototypes.
2 Open source JavaScript library for interactive web and mobile maps.
3 Open source JavaScript library for rendering web maps by using WebGL.
4 API for creating map tiles and designing web maps.

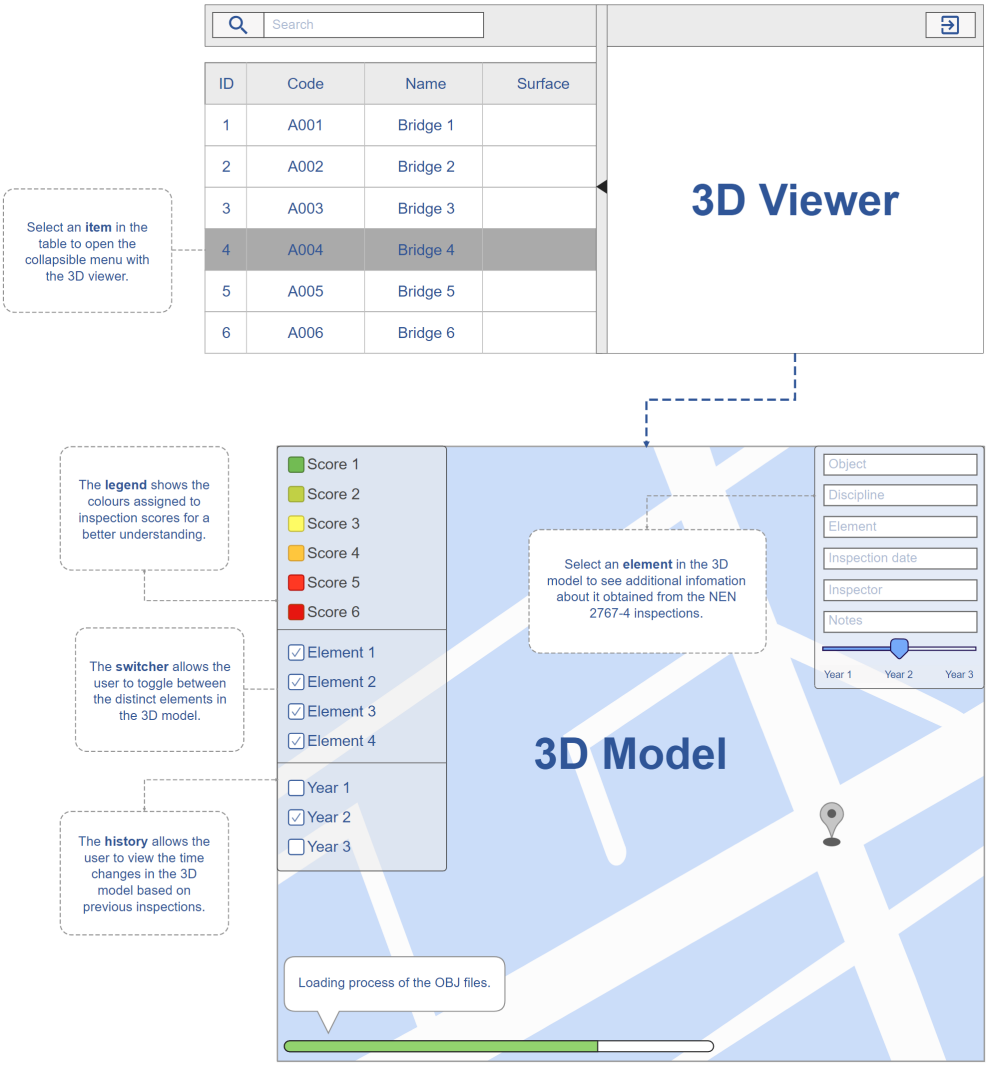


Figure 3.1: Mockup of the UI design of the 3D prototype.

In order to enable easier navigation and more user-friendly interface, the 3D viewer should be structured in several parts. When an element in the model is selected, a new tab should appear in the upper right corner to show additional information about the selected object, such as civil structure type, object name, inspection date, inspector, and notes. The information can be provided in the form of input fields, which are populated with data retrieved from the database. The fields should be editable so that the user can modify and update the information. In addition, a time slider can be incorporated in the information tab to give the user a better overview of the condition of the selected element in the past (see Figure 3.1).

A collapsible menu that contains a legend, switcher, and historical data can be added to the upper left corner of the 3D viewer. The legend aims to explain the inspection scores specified in the NEN 2767-4 standards and illustrate the colours assigned to them. A switcher can be represented in the form of check boxes so that the user can choose which elements in the 3D model should be shown in the viewer for further investigation. The historical data included in the menu should show previous results from the NEN inspections, i.e. the distinct elements in the BIM model should dynamically change their colours according to the inspection scores when the user selects a year. The years should be represented as radio buttons so that the user can choose only one item at a

time (see Figure 3.1).

Due to the large size of the BIM models, longer load times are expected. The long waiting period for the 3D model to load can be disturbing for the user, and thus, it is essential to add a progress bar on the bottom of the viewer (see Figure 3.1). The bar can inform the user how many of all files stored on the server are loaded.

The created UI design mockup was presented to the stakeholders from Sweco for evaluation. After the design concept has been accepted, the WebGL frameworks are reviewed to find a suitable technique for creating the prototype and proceeding with the development.

3.3 WEBGL FRAMEWORKS

The 3D graphics applications are usually developed by using high level programming languages like C or C++ and then compiled into an executable binary for a specific platform. This could entail complications in the usability of the applications on different operation systems and sharing the graphics due to the required plugins [Matsuda and Lea, 2013]. Recently, modern browsers are improving their rendering features by enabling the creation of interactive components through the use of the canvas presented in HyperText Markup Language (HTML) and interactive functionalities of JavaScript [Dirksen, 2013]. This allows the users to use the applications without installing any plug-ins in the web browser.

WebGL is a technology for drawing, displaying, and interacting with complex 3D graphics and animations by using HTML and JavaScript. It is a traditional built-in functionality supported by all modern web browsers since 2013 and does not require the installation of additional plug-ins and libraries. It enables the creation of enhanced online applications including 3D GIS solutions and virtual globes [Schilling et al., 2016; Matsuda and Lea, 2013].

The framework is developed based on Open Graphics Library (OpenGL), an open standard widely used in computer graphics and video games [Matsuda and Lea, 2013]. WebGL is compiled by the combination of JavaScript and OpenGL Shading Language Embedded Systems (GLSL ES), a shader program consisting of a vertex shader and a fragment shader [Eck, 2018]. This is the main difference between traditional dynamic HTML pages and HTML page with a WebGL content (see Figure 3.2).

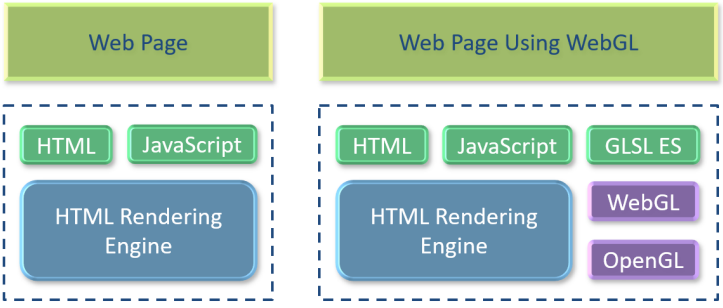


Figure 3.2: Software architecture of dynamic web pages and web pages using WebGL technology, adapted from Matsuda and Lea [2013].

There are various WebGL-based frameworks for the visualization of distinct 3D models and animations. Some of them are presented below to give a better overview of the existing web platforms, which are taken into account for the development of the 3D viewer.

3.3.1 Three.js

Three.js is an object-oriented JavaScript library for 3D graphics and animations, developed by Ricardo Cabello with contributions from other programmers. It provides a higher-level Application Programming Interface (API) for rendering 3D components built on top of WebGL, which makes the programming relatively easier than the traditional WebGL programming [Eck, 2018]. The library enables creating 2D sprite-based graphics and complex 3D geometries, animating and moving objects through a 3D scene, and applying various textures and materials to the objects. Moreover, Three.js supports a variety of file formats to load diverse BIM models, such as, JavaScript Object Notation (JSON), glTF, Wavefront OBJ, and COLLADA [Dirksen, 2013]. The WebGL library can be used for many purposes, mainly for simulations and game development (see Figure 3.3).

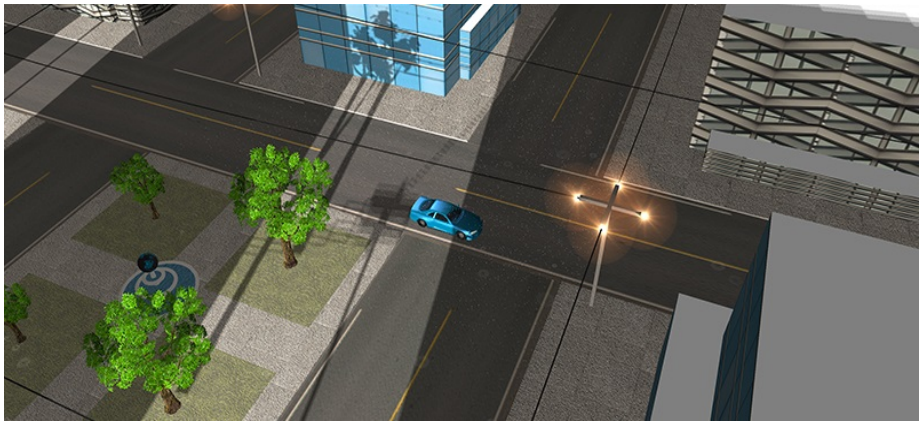


Figure 3.3: Customised application for city car driving built on Three.js [Poppe, 2017].

3.3.2 iTowns

iTowns is a WebGL framework for 3D geospatial data visualization, which is based on Three.js. It can represent the objects in globe or planar view depending on the purpose of the application [iTowns, 2017]. It was initially created by the Institut National de l'information Géographique et Forestière (IGN) to integrate 360-degree panoramic images, street images, 3D textured models, and point cloud data. Due to its ability to handle various geospatial data, the framework allows for developing advanced interaction functions, such as annotation, update plans, precise 3D measurements, simulation tools, virtual tours, and Augmented Reality (AR) applications [IGN, 2015]. However, the performance of iTowns is degrading in case of large-scale mapping, which is rather disadvantageous for the users.



Figure 3.4: Application for extruded WFS buildings on a virtual globe based on iTowns [iTowns, 2017].

3.3.3 Cesium

Cesium is an open source application for creating 3D globes and maps. The platform supports various spatial data sources and implements the major OGC web services like WMS and WFS, which makes it a viable option for building professional geospatial applications and creating 3D city models [Schilling et al., 2016]. Similar to Three.js, the most common BIM file formats can be loaded in Cesium, but they need to be converted to glTF supported by Cesium and rendered by using 3D Tiles. Besides the well-performing and precise maps, Cesium provides platform support and a large community to ease the development of the applications [Cesium, 2018].

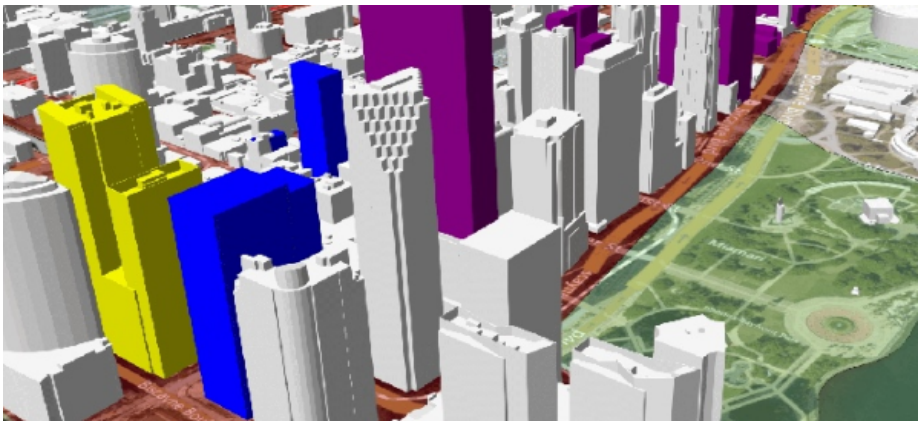


Figure 3.5: CyberCity 3D buildings visualized in Cesium for usage in planning and analysis, real estate, economic development, and flood risk mitigation [Cesium, 2018].

3.3.4 OSM Buildings

OSM Buildings is a JavaScript library for the visualization of 2D and 3D OpenStreetMap (OSM) building geometries on interactive maps. It provides great visual effects and a large amount of 3D objects. OSM Buildings is available in two versions: classic 2.5D and modern 3D. The classic version is characterised by a great performance and compatibility with all systems. It utilises Leaflet for the base map. The modern version is based on WebGL technology and Mapbox is used for the overview map. The web platform supports the GeoJSON and OBJ file formats [OSM Buildings, 2018].



Figure 3.6: 3D buildings visualized in OSM Buildings [OSM Buildings, 2018].

3.4 FRAMEWORKS COMPARISON

WebGL is a very powerful technology for web-based 3D visualization, which can be easily integrated in multidisciplinary applications. All the WebGL frameworks presented above are open source, editable, and supported by the modern browsers. In order to find the most suitable technology for the 3D prototype development, the platforms are assessed by testing the demos or any customised applications built on top of the given platform. Table 3.2 summarises the results of the frameworks comparison. Nevertheless, it is relatively difficult to evaluate the sample applications adequately, because their purpose is different than the objective of the 3D prototype. In most of the cases, simple 3D objects are represented with universal animation functionalities, whereas the visualization of detailed BIM models requires more advanced techniques.

Table 3.2: Comparison of the existing WebGL frameworks for 3D web visualization.

Requirements	Three.js	iTowns	Cesium	OSM Buildings
BIM Formats Support	✓	✓	✓	✓
Dynamic Colour Assignment	✓	✓	✗	✗
Subsurface View	✓	✓	✗	✗
Object Selection	✓	✓	✓	✓
Georeference Tool	✗	✓	✓	✓
Rotate, Zoom, Pan	✓	✓	✓	✓
Layers Control	✓	✓	✓	✓
Lightweight API	✓	✓	✗	✗
High Speed Loading	✓	✗	✗	✗

On the basis of the above stated platforms comparison, Three.js and iTowns possess the greatest possibility of being applied for the development of the 3D prototype. Cesium and OSM Buildings are not good candidates due to the lack of subsurface view and customisation for BIM models. Both WebGL frameworks are used for large-scale mapping rather than representing a single 3D model, which is the case of the desired

3D viewer. Furthermore, the virtual globes and maps are too heavy to be embedded in existing web applications like OBSURV and may affect their overall performance.

Three.js is widely used for the visualization of complex 3D models due to its extensive processing tools. Various BIM models can be loaded and customised according to the user preferences. Moreover, Three.js is a lightweight API and allows for easier integration into Web GIS systems than 3D globes and maps like Cesium and OSM Buildings. In that sense, Three.js fulfils all user requirements except the georeferencing, which in turns is the strength of iTowns.

As mentioned in Section 3.3.2, iTowns is fully based on Three.js and supports its built-in functionalities. In addition, it handles 3D geospatial data and provides the geolocation of the 3D objects. This is advantageous for the integration of a base map in the 3D viewer. However, the web platform does not always perform well when representing a larger area. During the platform evaluation, it could not be estimated whether the dynamic colour assignment of the BIM models will work as desired due to the lack of examples in the demos.

Both Three.js and iTowns are collaborative platforms with large communities of developers, which share their knowledge and expertise for solving different problems in the implementation. In addition, detailed documentations are provided for guidance in the development process. Nevertheless, Three.js is a more popular JavaScript library with many existing demos in various application fields and a greater support by its contributors.

To summarise, Three.js is chosen for the development of the 3D viewer. The following BIM file formats can be loaded in the library for visualization:

- **JSON**: Common file format for representing JavaScript objects as strings, which allows for data interoperability due to its ability to store and transfer structured data [Eck, 2018].
- **glTF**: Specification for the efficient transmission and loading of 3D scenes and models by applications by minimising the size of the 3D objects and run-time processing needed to unpack them [Khronos Group Inc., 2018b].
- **Wavefront OBJ**: Simple data format that represents the 3D geometries in a human readable format by specifying the position of vertices, vertex normals, and faces [Three.js, 2018].
- **COLLADA**: eXtensible Markup Language (XML)-based schema for 3D content exchange between applications, which provides comprehensive encoding of visual scenes, including complex geometries, shaders and effects, physics, animations, and kinematics [Khronos Group Inc., 2018a].

In this case, the BIM model used for the 3D prototype development should be either provided in one of the supported formats or converted to one of them during the data processing. The bottleneck of Three.js is the lack of georeferencing tools, which makes the integration of a base map in the 3D viewer more difficult. Therefore, it is necessary to find an alternative solution to align the 3D model with the base layer.

4 BIM MODEL PROCESSING

This chapter presents the BIM model provided for integration in an existing GIS application. For the development of the 3D prototype, it is important to get familiar with the model, which will be visualized in OBSURV. After explaining the dataset structure, a method for decomposition of the BIM model according to the NEN 2767-4 standards is described and the results of the decomposition are shown.

4.1 BIM DATASET

The BIM model of Koninginnebrug[1] in Rotterdam is used for the development of the 3D prototype (see Figure 4.1). The movable bridge is located over the Koningshaven port and connects the Noordereiland island with the southern part of Rotterdam. It was built in 1929 and is declared a national monument of great architectural, cultural-historical and construction-technical value. Koninginnebrug is a double bascule bridge with associated bridge guard houses and is characterised by solid curved lattice girders of riveted steel over a middle passage of 50 meters [Erfgoed, 2014].

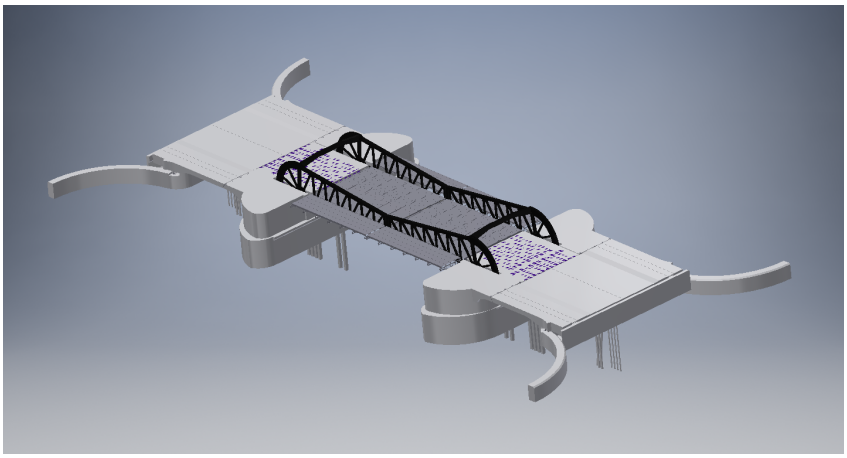


Figure 4.1: 3D model of Koninginnebrug.

The 3D model was created in Autodesk Inventor, a BIM modelling platform for product and mechanical design, dynamic simulations and rendering [Autodesk, 2018]. The model was developed in LOD 300, in which the 3D object is graphically represented and its quantities, size, shape, and orientation are specified and measurable (see Section 2.3.1). The BIM model was provided in STEP format, an International Organisation for Standardisation (ISO) 10303 standard for digital product data representation and exchange. The standard focuses on the visualization of 2D and 3D geometry models, geometrically bounded surface geometry models, topologically bounded surface geometry models, compound shape geometry models, constructive solid geometry

1 Koninginnebrug = Queen's Bridge (engl.)

models, etc [ISO10303-242, 2014].

In Autodesk Inventor, STEP models are handled as assemblies with a specific structure, in which the main assembly can contain several assemblies and subassemblies composed of solid parts. This hierarchical structure is beneficial in particular for data processing and decomposition. As shown in Figure 4.2, the initial BIM model of the bridge was structured in four assemblies:

- **Abutment:** Location where a bridge reaches the ground and joins the road. The abutment should be placed far enough to ensure a decent side walk below the bridge and prevent from offsets, which can influence the bridge maintenance [Gottemoeller, 2014].
- **Approach slab:** Concrete transitional roadway between the pavement and the bridge deck, designed to reduce the dynamic effects of the vehicles on the bridge. The approach slabs require particular attention for inspection and maintenance due to the increasing traffic [Masirin and Zain, 2013].
- **Pier:** Structural frame consisting of circular or rectangular columns with a cap beam that supports the bridge superstructure. The pier placement is crucial for the visual representation of the bridge structure and its surrounding topography [Gottemoeller, 2014].
- **Bridge deck:** Superstructure element, which includes the concrete slab, overhangs, railings, parapets, and provisions for drainage [Gottemoeller, 2014].

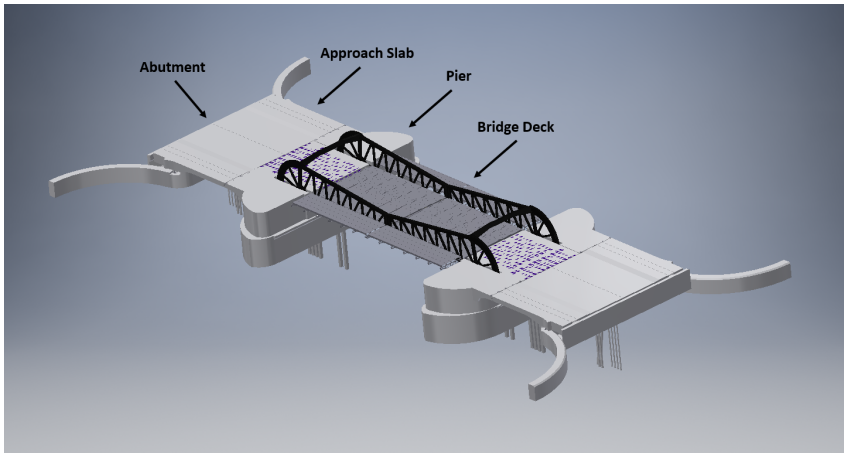


Figure 4.2: Composition of the in the initial 3D model in Autodesk Inventor.

Each of these bridge elements contains multiple subassemblies, which are grouped based on their geographical position. The abutments, approach slabs and piers have identical structure, including expansion bearings, pavement beams and construction beams. The bridge deck is composed of ballast boxes, counter weight and locking installation, which support the bridge opening, as well as driveway and footpath for the cars and pedestrians (see Appendix A).

4.2 BIM DECOMPOSITION

In order to use the 3D prototype for inspection and maintenance of civil constructions, it is necessary to simplify the detailed BIM model and represent the bridge elements as

specified in the NEN 2767-4 standards. The initial 3D model is relatively generalised by grouping the assemblies and subassemblies, but the composition does not fully comply with the NEN 2767-4 standards. Therefore, an appropriate decomposition of the given BIM model is required. The generalisation is made in Autodesk Inventor due to its functionality to simplify BIM models and convert the modified parts to various CAD formats like Wavefront OBJ, which is required for rendering in WebGL.

The NEN standardisation differentiates between fixed and movable bridges, as the latter include a great amount of components and require a more complex design and construction. The movable bridges are composed of several mechanical systems and structural components, such as a retaining structure, substructure, bascule cellar, superstructure, handrail structure, expansion joints and span lock (see Appendix B). However, the initial BIM model does not include all bridge elements, e.g. the bascule cellar located inside the superstructure, the expansion joints attached to each bridge element, and the handrail structure next to the footpath. Figure 4.3 illustrates the existing elements considered in the desired bridge decomposition:

- **Retaining structure:** Geosynthetic soil structure used to support the bridge abutments and approach roads [Skinner and Rowe, 2004].
- **Substructure:** Set of components, which support loads from the superstructure and transfer loads to the ground. This element consists of bearings, support blocks, abutments, piers, foundations and retaining structures [Hida, 2014].
- **Superstructure:** Structural system used to support the bridge, which includes the main horizontal spanning members like continuous girders, arches, trusses, bridge deck overhangs, parapets and railings, as well as counter weights [Gottemoeller, 2014; Koglin, 2003].
- **Span locks:** Elements to connect the tip ends of the bascule leaves, which should deflect equally to ensure compatibility of the deck deflections from the opposing span leaves under traffic loads [Catbas et al., 2013].

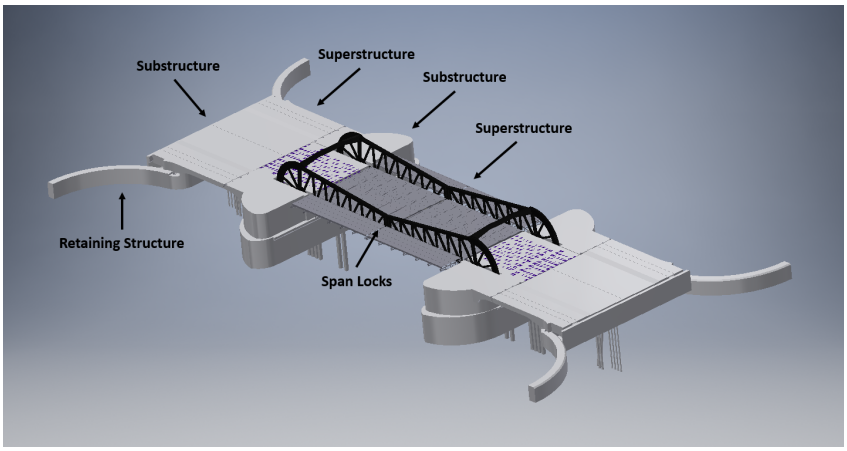


Figure 4.3: Decomposition of the 3D model according to the NEN 2767-4 standards.

In the initial BIM model, the retaining walls and the abutment are structured in the same assembly. For the decomposition, these building parts are separated by using the **Split** tool in Autodesk Inventor. The fragmentation is done by drawing a 2D reference line on the part face to identify where the part should be split into two solid bodies. The line is positioned in the beginning of the wing walls in the 3D model (see Figure 4.4).

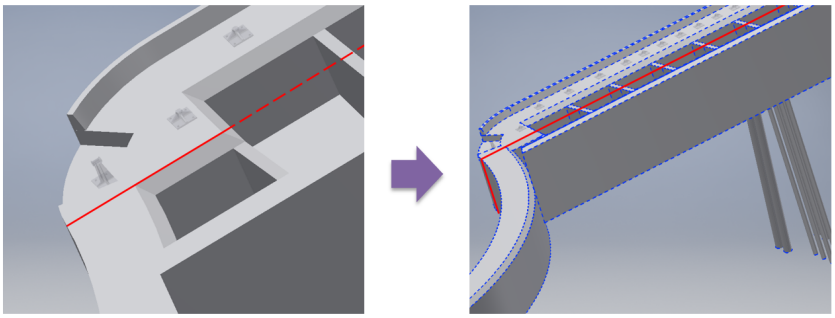


Figure 4.4: Splitting of the abutment part.

The edited abutment is stored in two separate assembly files for simultaneous manipulation. In the first file, the faces of the abutment part are deleted from the assembly to obtain the retaining structure and the remaining faces of the wing walls are saved as a simplified part in the Inventor Part (IPT) format. In the other file, the faces of the retaining walls are deleted from the assembly to retrieve the abutment (see Figure 4.5).

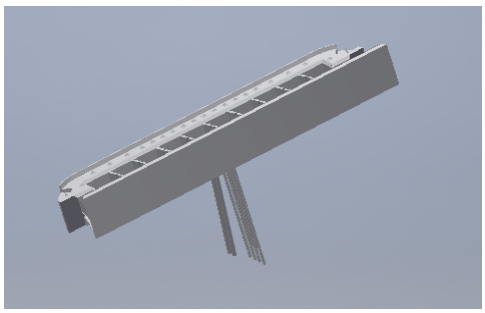


Figure 4.5: Decomposed assembly of the bridge abutment.

As the splitting tool divides the entire assembly into two solid bodies, the components of the edited abutment part should be combined into a single solid body. However, this is a time-consuming process, which involves the manual selection of each component. Thus, this element remains divided. For the construction of the bridge substructure, the modified abutment and the existing pier assemblies are combined into a single IPT file.

Similar to the substructure composition, the approach slab and bridge deck assemblies are merged into a simplified part to generate the superstructure. In the provided BIM model, the bridge deck contains the gears and bolts, which belong to the span lock element according to the NEN standards (see Figure 4.6). For that reason, the locking installation components are deleted from the superstructure element and stored in a separate IPT file.

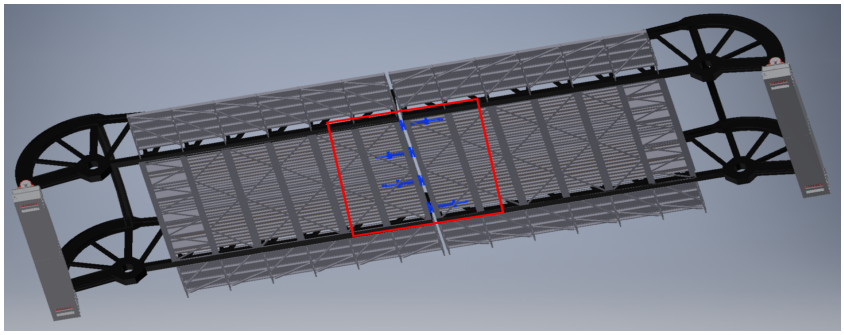


Figure 4.6: Span lock components in the bridge deck composition.

Another obstacle in the bridge decomposition is that the pivoting point is attached to the beams, while it should be a separate bridge element (see Figure 4.7). In that case, the pivot should be removed from the superstructure and stored as an individual assembly. This cannot be achieved at that stage of the project, as the component should be created again by drawing a 2D sketch and extruding it for designing its shape. Then, the topological relationships should be determined to obtain the correct location of the solid body in the 3D model.

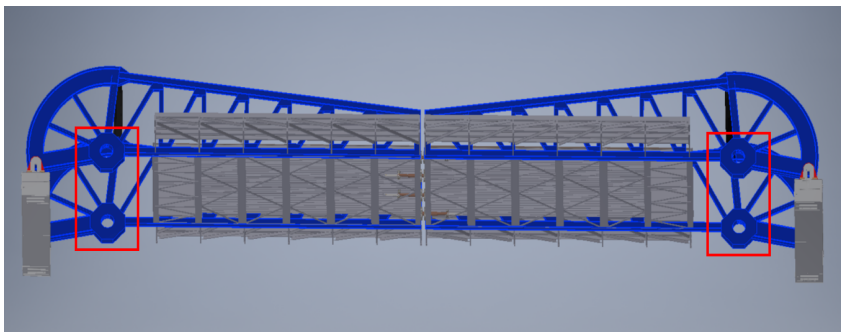


Figure 4.7: Pivoting point composition.

As a result of the data processing, the provided BIM model was restructured in four elements representing the main construction and electromechanical structures of the bridge. The same LOD 300 is preserved for all assemblies, as the decomposition process required a rearrangement of the geometries only (see Figure 4.8).

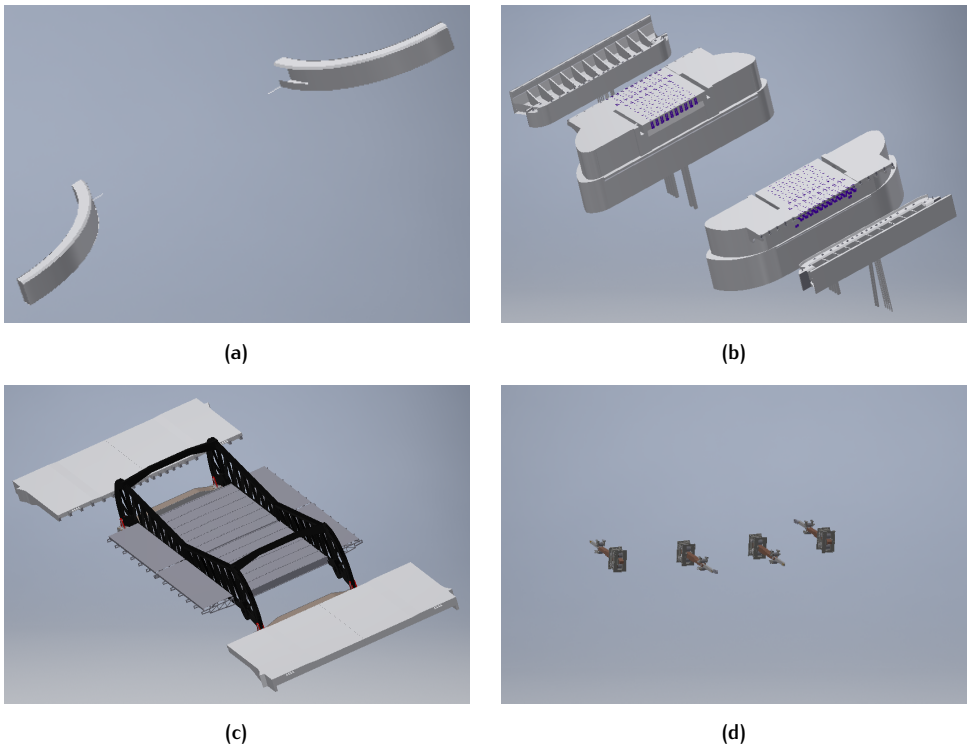


Figure 4.8: Bridge decomposition: (a) Retaining structure. (b) Substructure. (c) Superstructure. (d) Span locks.

Each element is stored in simplified parts and directly converted to OBJ as one of the few formats suitable for rendering in Three.js from all BIM file formats supported by Autodesk Inventor. The OBJ file is extensively used for sharing 3D models because of its import and export support from various 3D modelling platforms. The file format

encodes the surface geometry of 3D objects represented as triangles, quadrilaterals, polygons, or free-form curves. In addition, the color and texture of the 3D model are specified in an accompanying MTL file [Chakravorty, 2018].

5 PROTOTYPE DEVELOPMENT

Having presented the UI design of the prototype and processed the provided BIM model according to the NEN 2767-4 standards, the development of the desired prototype is explained. At the beginning, the test environment is prepared by creating a custom module for civil structures in OBSURV and integrating a collapsible page for the 3D viewer, where the BIM model will be visualized. After that, the viewer is configured in Three.js and the functionalities illustrated in the UI design concept are implemented.

5.1 TEST ENVIRONMENT PREPARATION

The 3D viewer is integrated into a custom module for civil constructions as part of the OBSURV application package. The test environment is created in Oracle APEX by taking the example of an existing module for public lighting. APEX is a low code tool for building database-driven web applications on-premise or in the cloud in a single, extensible platform. Furthermore, APEX manages all important aspects in the application development, such as security, session state management, concurrency, data persistence, responsive design, and more [Oracle, 2018]. The APEX application pages are handled as HTML pages, i.e. each APEX component can be assigned an HTML tag in order to add Cascading Style Sheets (CSS) styling and JavaScript interactions.

APEX provides a specific tool to connect Oracle databases and manage the objects. The tool allows for creating and viewing object properties, entering various SQL commands, storing and running scripts, and loading text and spreadsheet data [Godfrey, 2017]. The Oracle database for civil constructions, which contains the test data of Rotterdam, is imported in the custom module. The database includes the tables for the public assets and their elements, inspections, and condition status, which are used in the 3D prototype development (see Appendix C).

The civil constructions table contains essential information about the assets, their construction year, width, height, surface area, construction material, as well as their owner, manager, and constructor. Each civil construction is composed of several elements, which are stored in a separated table. The elements are connected to their corresponding civil construction object by using its unique ID. The results from the NEN 2767-4 inspections reported in the inspection reports are saved in the inspections table, which is linked to the tables of the civil constructions and their elements in order to assign the inspection information to the correct civil construction. The status table is a lookup table, which includes the condition scores assigned to the civil constructions during the inspections and description of the scores.

After arranging the database, the custom module for civil constructions is customised by adjusting several application pages and components in Oracle APEX with the

Application Builder tool. The login page configured for the existing public lighting module is preserved, as the same page is required for the new module as well. The form includes input fields for the user name and password to allow the users to add their credentials and a submit button for accessing the application.

The APEX application pages are composed of various elements, such as regions, lists, and interactive reports. In order to make the application more user-friendly, the pages are grouped into sections added to a dynamic navigation menu for an easier access. Each OBSURV module contains inventory, inspection, budgeting, planning and reporting sections, which the asset managers can use for different purposes.

The prototype is developed in the inventory section, which includes an interactive report with the civil structures in Rotterdam retrieved from the associated table in the database by using an SQL selection query. Interactive reports enable the users to view and customise the data appearance through searching, filtering, sorting, and highlighting [Jennings, 2013]. Furthermore, two buttons are added to the interactive report to allow the users to create new entries in the table and to make selections (see Figure 5.1).

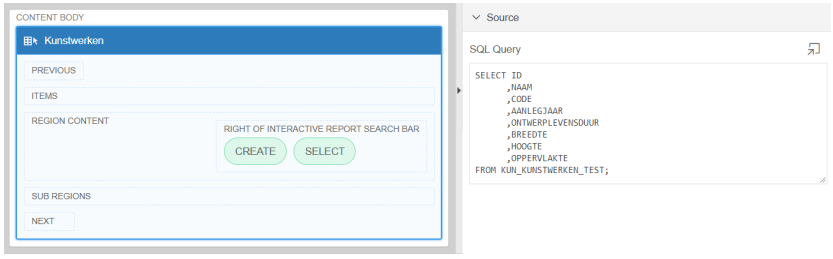


Figure 5.1: Configuration of interactive report of civil constructions.

Figure 5.2 illustrates the home page, which directs the user to the table of civil constructions stored and some of their attributes. As the entire dataset of civil structures in Rotterdam is very large, only a few bridges are included in the table to avoid long loading time.

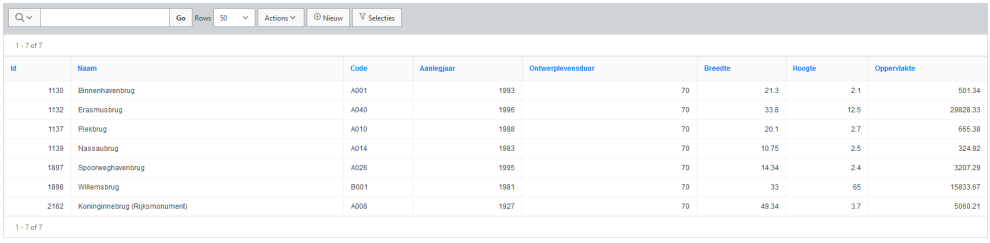
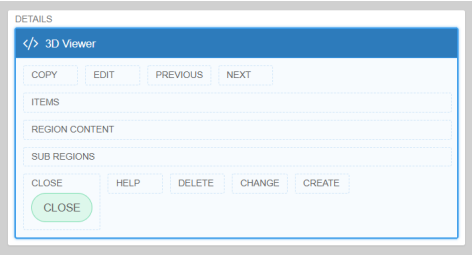


Figure 5.2: Home page of the custom module.

A separate collapsible page is created for the 3D viewer so that it can be opened when an object in the interactive report is selected. This functionality is configured in jQuery by assigning HTML tags to the APEX components and using the **click** event. It retrieves the user selection from the table and removes the collapse property of the details page in order to render its content. Furthermore, a button is added to close the details page and return to the interactive report (see Algorithm 1).



Algorithm 1 Configure collapsible page

```
function CLICK( table )
    add is-selected attribute to selected object
    expand details page for selected object in table
    scale rendering theme
function CLICK( button )
    remove object selection in table
    collapse details page
```

Next, the 2D objects in the interactive report are linked to their corresponding 3D models. The OBJ files are stored in the directory of the custom module on the server side due to their large size. The file names contain the unique ID of the bridge in the database, the name of the bridge element, and the *obj extension. The files are defined as strings and added to a JavaScript array for easier retrieval in the rendering process (see Section 5.2). The ID of the selected object is obtained from the interactive report and included in the file name string (see Algorithm 2). This ensures that the correct BIM model is shown in the details page.

The 3D prototype is developed by using one 3D model. Therefore, the viewer should be visible only for the associated 2D object and remain empty for all other objects, as there are no files in the specified directory. The HTTP response status is used to identify if the request is successfully completed or errors occur during the files loading process. The most commonly used statuses are **200 OK** for successful responses and **404 Not Found** when the server cannot find the requested resource [Mozilla, 2018b]. The XMLHttpRequest object is implemented to check the HTTP status for each OBJ file due to its ability to send and receive data from the server. A specific function is executed whenever the status of the request changes to inform about the request progress [W3Schools, 2018]. The HTTP status 404 refers to a client error response, as the server cannot find the requested resource [Mozilla, 2018b]. The 3D viewer is hidden in case the HTTP status is 404 and otherwise the HTML element, which contains the Three.js rendering functionality is loaded (see Algorithm 2).

Algorithm 2 Associate 2D objects with their corresponding 3D model

```
function CLICK( table )
    id ← ID of selected object in table
    files ← array containing the names of the OBJ files
    ▷ construct the OBJ file name as follows:
    "path/to/folder/" + id + "element name.obj"
    for each file in files array do
        xhr ← XMLHttpRequest object to request data from server
        function ONREADYSTATECHANGE( xhr )          ▷ executed when HTTP status changes
            if HTTP status = 404 then
                make 3D viewer content invisible
            else
                make 3D viewer content visible
        initialize new request
        send request
```

The 3D viewer is developed by combining HTML, CSS, and JavaScript. The HTML code is added as source text to determine the details page content. The CSS styling file and all JavaScript libraries required for rendering the 3D model are added separately to the **File URLs** settings of the application page.

5.2 3D VIEWER CONFIGURATION

5.2.1 Scene Creation

The 3D viewer is built completely on Three.js, which creates a scene graph out of a camera, lights, and graphical objects and animates the represented scene [Eck, 2018]. For that purpose, it is required to configure a WebGL renderer, add 3D objects, and enable an animation (see Figure 5.3). The renderer is used to create an image from the scene graph. The 3D objects are made up of geometry and material to make the object visible in the scene [Eck, 2018]. The animation system integrated in Three.js allows the renderer to update the camera and draw the scene every time the screen is refreshed [Three.js, 2018].

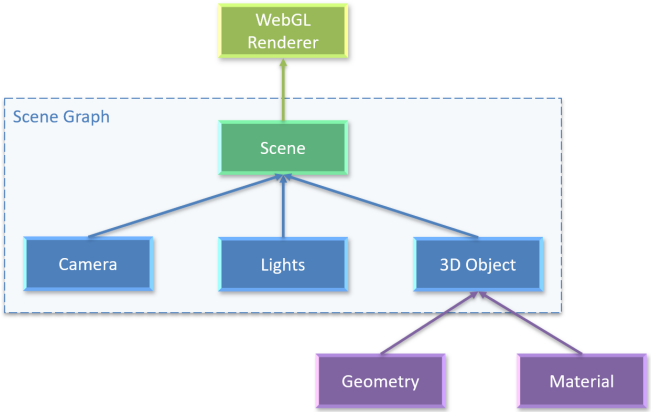


Figure 5.3: Rendering process in Three.js, adapted from Taalman [2015].

The WebGL renderer for displaying the created graphics and animations is positioned inside the HTML element defined for 3D rendering, i.e. the renderer size and view port are set according to its width and height. The background colour is white to align with the other components of the 3D viewer and a scene is added to place the projection camera and lights (see Algorithm 3).

In the next step, the camera is configured. The perspective camera is chosen, as it is most commonly used for 3D rendering in computer graphics. The perspective projection is designed to imitate the view of the human eye where the objects appear smaller at a distance [Three.js, 2018]. The camera has several properties, such as field of view, aspect, and near and far plane, which should be taken into account in the configuration (see Figure 5.4).

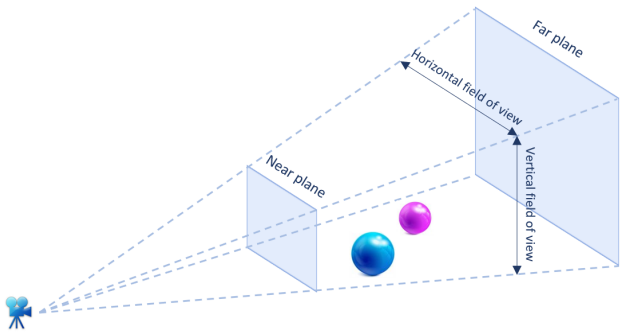


Figure 5.4: Perspective camera model, adapted from Dirksen [2013].

The vertical field of view is set to 70° and the aspect ratio is computed as the ratio between the WebGL renderer width and height so that the 3D objects do not look compressed on the screen. The near and far planes determine the start and end points of the drawing process, i.e. objects further away from the camera than the value of far plane or closer than the value of near plane are not rendered in the scene [Three.js, 2018]. The perspective camera is located in the center of the scene at a larger distance so that the 3D model does not fully cover the WebGL renderer (see Algorithm 3).

Lighting is a fundamental aspect in computer graphics, as it provides a more realistic representation of the 3D model by simulating light sources and their interaction with the objects in the scene. Three.js offers various lighting options, such as ambient, directional, hemisphere, and point lights. The ambient light is commonly used in 3D rendering due to its ability to illuminate the 3D model equally by using light, which has been reflected and re-reflected many times without having a particular direction [Eck, 2018]. Therefore, the ambient light is implemented in the 3D viewer and a white colour is assigned to it to make the 3D objects appear brighter (see Algorithm 3).

Consequently, orbit controls are added to the 3D viewer to control the perspective camera motions and enable zooming and rotating the 3D model. The maximum zoom is set to the same value as the camera far plane to avoid the disappearance of the object when zoomed in further. In case the orbit controls are updated, a JavaScript function is executed to render the 3D object as requested and update the animation (see Algorithm 3).

Algorithm 3 Configure camera, lights, and animation controls

```
function INIT
    webglCanvas ← HTML canvas with WebGL content
    renderer ← WebGL renderer with the size of the HTML canvas
    add renderer to the HTML canvas and make it visible
    add a mousedown event to the WebGL canvas
    add a resize event to the WebGL canvas
    scene ← Three.js scene

    camera ← perspective camera with X, Y, Z position added to the scene
    light ← soft white ambient light added to the scene
    controls ← orbit controls to enable zooming, rotating, and panning
    add a change event to the controls
```

5.2.2 BIM Model Loading

After the main components of the Three.js scene are set up, the 3D model is loaded in the WebGL renderer by using the OBJ loader to retrieve the OBJ files from the server. By default, the objects are represented as meshes with predefined geometry and material, which can be modified if any changes in the 3D model are required [Three.js, 2018]. Each object is inserted into a predefined JavaScript array, which can be used for other computations in a later step (see Algorithm 4). In addition, a loading manager is incorporated for the OBJ loader to track its progress and display it in the console. The manager is an optional functionality in Three.js, but its use is highly recommended due to its ability to record the loaded and pending data and show if any errors occurred during the loading process. In case of the 3D viewer, it is used to configure the progress bar to display how many files have loaded successfully (see Algorithm 4).

Algorithm 4 Load the OBJ files in the scene

```
manager ← loading manager
progress bar width ← percentage of loaded files

index = 0
objects ← empty array
function LOADFILE( scene, info )
    loader ← OBJ loader supported by Three.js
    function LOAD( object )                          ▷ iterating over the files array
        object.userData.Name ← last string extracted from the file name
        add loaded object to the scene
        add loaded object to the objects array
        index ← index ++
        call loadFile procedure
```

In order to colour the distinct bridge elements according to the results of the NEN inspections, the inspection data is retrieved from the database by using Asynchronous JavaScript and XML (AJAX) calls. The **userData** property, enabled for each object in Three.js, is used to store the retrieved information while loading each OBJ file (see Algorithm 4).

Oracle APEX provides **on demand** application processes, which can be used to perform various actions required for the application pages. The processes utilise PL/SQL to send to and retrieve information from the database and are executed at specific point in the page or from an AJAX callback [Oracle, 2015]. The data retrieval is a stepwise process, which includes defining a PL/SQL procedure, specifying an APEX application process, and obtaining an AJAX callback response (see Figure 5.5).

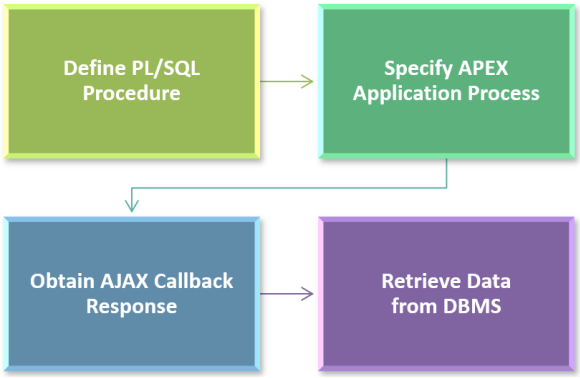


Figure 5.5: Data retrieval by using AJAX calls in APEX.

First, a PL/SQL procedure is created within a custom package for inspections of civil constructions in Oracle SQL Developer (see Algorithm 5). Then, the database fields of interest are declared and their type is obtained from the inspections table. The unique ID of the civil construction element is required to link the bridge element to the database entry. The inspection date and score, inspector name, and additional notes will be used in the object selection in a later step. A bulk collection is used to retrieve all table rows with a single fetch and improve the speed of the data retrieval [Feuerstein, 2012]. The response from the PL/SQL procedure can be stored in a simple JSON object as a common data type for information exchange information accessible from JavaScript [Petrus, 2013]. The JSON object is created with the **htp.p** command, which generates an HTML paragraph tag with the specified content [Oracle, 2005]. Each database record

is added in a separate JSON object and all objects are combined in a JSON array for easier retrieval.

Algorithm 5 Configure the PL/SQL procedure to retrieve data from Oracle

```
create package KUN_INSPECTIES_APEX
create procedure GET_INFORMATION and define datatype of the fields to be retrieved
declare nested table KUN_REC_TT and table type based on the inspections table
define collection KUN_REC_T holding the IDs of all inspections
define index KUN_IDX of created table as integer
begin
    use bulk collection to fetch all inspection IDs into KUN_REC_T
    add inspection IDs to the collection          ▷ iterate over IDs to check if KUN_IDX ≠ 0
    add element ID to JSON object
    add inspection date to JSON object
    add inspector name to JSON object
    add score to JSON object
end
```

After defining the PL/SQL procedure to retrieve information from the database, an application process is created in APEX with the same name as the procedure name. An AJAX callback is chosen to run the process when requested and the PL/SQL procedure is obtained from Oracle SQL Developer to set up the application process. In Three.js, a separate function is added to call the PL/SQL procedure (see Algorithm 6). This function is a wrapper for the **jQuery.ajax()** function to perform an asynchronous AJAX request. All the jQuery settings and additional APEX features are provided to specify the loading indicator position and the data type of the AJAX callback response. The function also has a success handler, which is executed when the call is completed, and the argument **pData** refers to the response sent from the AJAX request parsed in the form of the defined data type [Petrus, 2013]. The final JavaScript arrays with the database records are used in the **loadFile** function for loading the 3D model.

Algorithm 6 Retrieve data from database and add it to the model properties

```
function GETINFORMATION( objectLoader, scene )
    get_information ← APEX server process
    loadingIndicator ← HTML body
    loadingIndicatorPosition ← HTML page
    dataType ← text
    function SUCCESS( pData )
        info ← parsed JSON array with retrieved data

function LOADFILE( scene, info )
    ...
    for each object in objects array do
        object.userData.DatabaseID ← add object ID
        ▷ data extracted from the info array
        object.userData.InspDate12 ← add inspection date for 2012
        object.userData.Inspector12 ← add inspector name for 2012
        object.userData.Score12 ← add score for 2012
        repeat for the other inspection years
    ...
```

For assigning the initial colour to the objects, it is necessary to traverse each 3D object and change the material of its components, which is done by using the **colorComponent** function. The colour depends on the condition score specified for the bridge element during the NEN inspections and the colour range is defined according to the documentation of the inspections. The material is made transparent and slightly opaque for a

better representation in the scene (see Algorithm 7). The initial colours of the BIM model are obtained from the last bridge inspection in 2012.

Algorithm 7 Assign initial colour to the model elements

```
function LOADFILE( scene, info )
    ...
    for each object in objects array do
        traverse the object components
        call colorComponent to colour the components
    ...

function COLORCOMPONENT( component, score )
    if score = 1 then
        assign green colour to the component
    else if score = 2 then
        assign light green colour to the component
    else if score = 3 then
        assign yellow colour to the component
    else if score = 4 then
        assign orange colour to the component
    else if score = 5 then
        assign red colour to the component
    else
        assign dark red colour to the component
```

Figure 5.6 illustrates the BIM model visualized in the WebGL renderer and coloured based on the condition score specified in the NEN inspections. A legend is also incorporated on the upper left corner to explain the meaning of the colours in the 3D model for easier interpretation of the model.

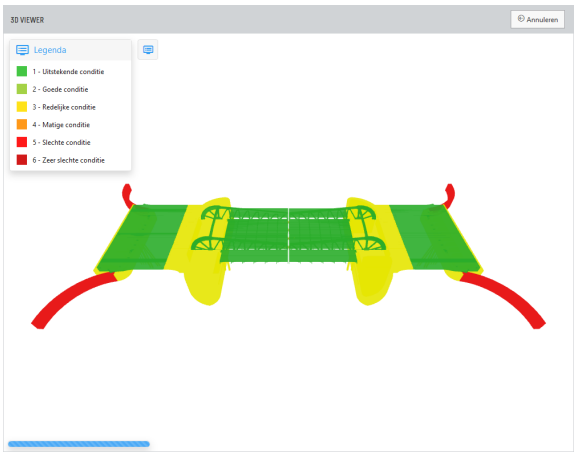


Figure 5.6: 3D viewer with the integrated BIM model and legend.

5.2.3 Base Map Configuration

GIS can provide a deeper insight into the BIM technology by enabling advanced analysis and visualization of the surrounding environment of 3D objects. However, this is possible only when the 3D models are properly located on a geographic map [Diakite, 2018]. There are various options for incorporating a base map in the 3D viewer. The most common web mapping tools are Leaflet, OpenLayers, and Mapbox. However, a larger map can affect the performance of the web application and does not allow the user to rotate the map and view subsurface objects, e.g. the bridge substructure in case

of the BIM model.

The alternative is to add a square plane below the 3D model in Three.js and apply a texture to it in the form of an image of the surrounding area. In order to define the bounding box of the plane, it is necessary to specify its X and Y lengths. This can be achieved by obtaining the parameters of the outer element of the BIM model with the largest spatial extent - retaining walls (see Algorithm 8).

Algorithm 8 Determine bounding box of the ground plane

```
index = 0
function LOADFILE( scene, info )
    function LOAD( object )
        ...
        if index = 0 then                                  ▷ OBJ file of the retaining walls
            call makeBBox function to define bounding box of the given object
        ...

function MAKEBBOX( object )
    bbox ← set bounding box from object
    xLength ← define x length of bounding box
    yLength ← define y length of bounding box
    zLength ← define z length of bounding box
```

The texture of the plane is obtained by sending a WMS request to the PDOK server to receive an image of the area of interest. First, it is required to define the coordinates of the center of the area in the Dutch CRS, which is a 2D Cartesian reference system and uses meters as units of measurement. For that purpose, the coordinates of the centroid of the 3D model are to the desired CRS and the lengths of the bounding box are converted in meters, as the coordinates in the OBJ files are given in inches. Then, the lower left and upper right corners of the WMS image are calculated with respect to the center point. The WMS request is created by accessing the PDOK topographic map *TOP10NL*, choosing the layers and their styles, and defining the spatial extent.

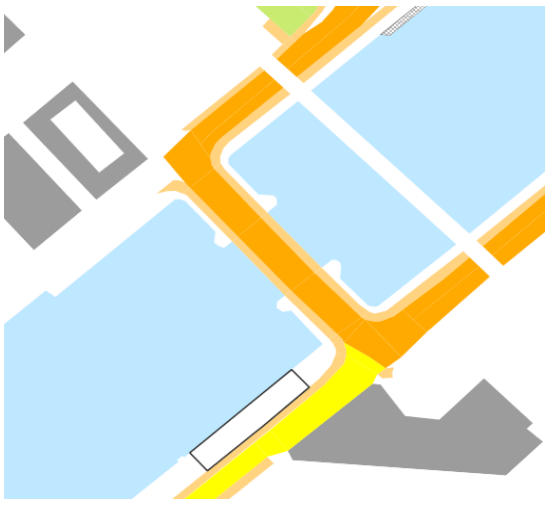


Figure 5.7: Image of the area of interest, obtained as a WMS request.

In the end, the ground plane is constructed with the predefined geometry and the resulting PNG image is loaded as a texture. In addition, a toggle button is added to the 3D viewer to allow the users to switch the ground plane on and off based on their own

preferences. The button is represented as a check box and the ground plane is made visible or invisible depending if the check box is clicked or not (see Algorithm 9).

Algorithm 9 Create ground plane to place base map

```
centerX ← X coordinate of bridge centroid in Dutch CRS
centerY ← Y coordinate of bridge centroid in Dutch CRS

function CREATEGROUND( valuesBBox, objectRotation )
    lengthMeters ← convert xLength and yLength to meters
    leftX ← define left side of the ground plane
    lowerY ← define bottom of the ground plane
    rightX ← define right side of the ground plane
    upperY ← define top of the ground plane
    baseURL ← specify base URL for the WMS request
    layers ← choose layers for the ground plane
    style ← choose style for the layers
    wmsRequest ← create WMS request with the defined parameters
    groundMaterial ← load image from WMS request as texture
    groundGeometry ← get xLength and yLength as boundaries
    add ground plane to the scene

function SHOWGROUND
    if togglebutton is clicked then
        make ground plane visible
    else
        make ground plane invisible
```

In the next step, the 3D model needs to be aligned with the ground plane. BIM models are rarely associated with real-world coordinates, which is why it is important to obtain their exact geographic location on Earth for integration in a GIS system [Diakite, 2018]. The BIM models are developed in a 3D Cartesian CRS, which is used to describe the position of objects in a 3D space given in X, Y, Z coordinates. The CRS is defined by the origin of the axes, which is selected arbitrarily, and their direction. In the traditional Cartesian CRS, the X and Y axes are in the forward and backward directions and the Z axis is orthogonal to the other two [Marel, 2014]. However, many BIM design platforms like Autodesk Inventor utilise a User Coordinate System (UCS), a specific type of a Cartesian CRS that has the Y axis in the up direction instead of the Z axis [Autodesk, 2014]. The two types of Cartesian reference systems are illustrated in Figure 5.8 below.

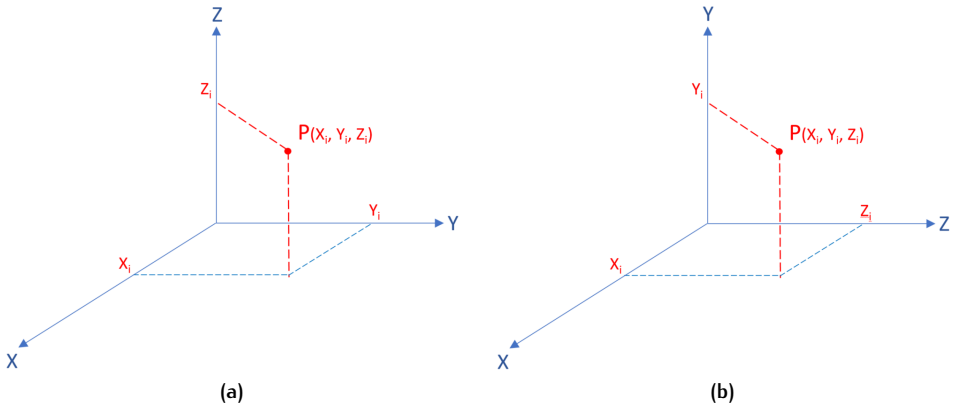


Figure 5.8: Cartesian CRSs: (a) Traditional reference system. (b) UCS reference system.

As explained in Section 2.4, there are different methods for georeferencing 3D models. A combined framework based on the approaches proposed by Kolár and Wen [2009] and Wang et al. [2013] is developed and applied to the decomposed BIM model. The first step of the georeferencing process is to define several reference points on the bridge and obtain their geographic location by using a cross-platform like Google Earth. The chosen points of interest are the centre, north-eastern most and south-western most points of the bridge superstructure (see Figure 5.9).

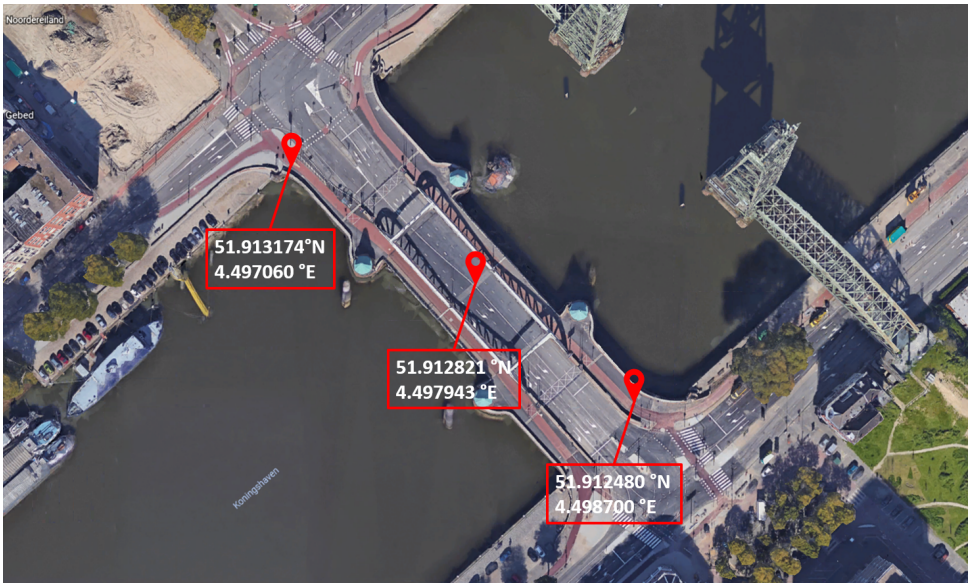


Figure 5.9: Reference points and their coordinates obtained from Google Earth.

The coordinates provided in Google Earth are in World Geodetic System 1984 (WGS84), a global terrestrial reference system used by the Global Positioning System (GPS) to determine positions on Earth. WGS84 represents the Earth surface as an oblate ellipsoid, which is considered to be the best approximation of shape of the Earth. The coordinates are given in latitude φ, longitude λ, and height h above or below the reference ellipsoid [Marel, 2014]. As can be seen in Figure 5.9, the location of the points of interest is expressed only by the latitude and longitude and the ellipsoidal height is not taken into account in the georeferencing, as it is not of great importance for the coordinate conversion.

The WGS84 ellipsoid can be defined by several parameters, such as the equatorial radius a (6 378 137 m) and polar radius b (6 357 m) that determine the flattening f and eccentricity e Marel [2014]:

$$f = \frac{a-b}{a} \qquad e^2 = 2f - f^2 \tag{5.1}$$

In the next step, the geographic coordinates (φ, λ, h) are converted to 3D Cartesian coordinates (X, Y, Z). The resulted coordinates are in an Earth-Centered, Earth-Fixed (ECEF) reference system, a global CRS with an origin at the center of the Earth (see Figure 5.10). ECEF is characterised by X and Y axes in the equatorial plane and Z axis orthogonal to the other two [Marel, 2014; Royal Observatory of Belgium, 2012]. The geocentric CRS is usually used in navigation systems to determine the satellite and receiver positions [Sanz Subirana et al., 2011b].

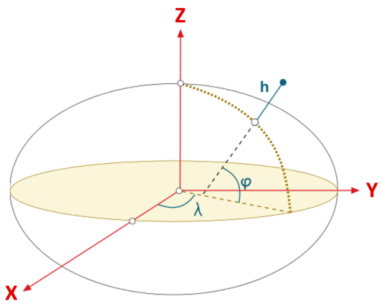


Figure 5.10: Schematic representation of the WGS84 and ECEF coordinates [Sanz Subirana et al., 2011a].

The ECEF coordinates can be obtained from the ellipsoidal coordinates by using the following equations:

$$\begin{aligned} X &= (\bar{N} + h)\, cos\varphi\, cos\lambda \\ Y &= (\bar{N} + h)\, cos\varphi\, sin\lambda \\ Z &= (\bar{N}\,(1 - e^2) + h)\, sin\varphi \end{aligned} \tag{5.2}$$

where $\bar{N}$ refers to the radius of curvature related with the eccentricity:

$$\bar{N} = \frac{a}{\sqrt{1 - e^2\, sin^2\varphi}} \tag{5.3}$$

as proved by Sanz Subirana et al. [2011a]. However, ECEF stores up to 7 digits before the decimal point per axis, which is inconvenient and differs from the coordinates specified in the OBJ files. Therefore, the ECEF coordinates need to be converted to local coordinates related to the centroid of the 3D model. The local CRS is often referred to as East, North, Up (ENU) reference system, in which X points in the East direction, Y in the North direction and Z in the up direction (see Figure 5.11). Both the geocentric and topocentric CRSs are tied to the Earth, but their origins differentiate depending on the use case [Marel, 2014].

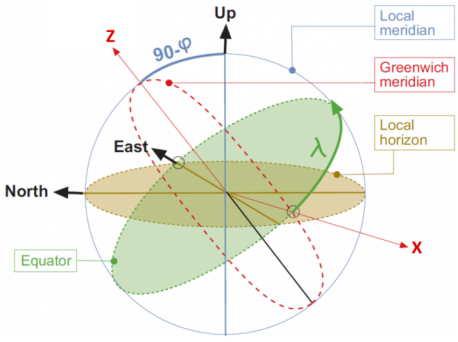


Figure 5.11: Schematic representation of the ECEF and ENU coordinates, adapted from Sanz Subirana et al. [2011a].

In the Cartesian coordinates conversion, the datum of the target CRS should be changed and the coordinate axes should be defined by using a 7-parameter similarity transformation: a scale parameter, 3 rotation parameters, and 3 translation parameters [Marel, 2014]:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = (1+\mu) \cdot R(\Omega_x\ \Omega_y\ \Omega_z) \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{5.4}$$

where *X'*, *Y'*, *Z'* are the ENU coordinates and *X*, *Y*, *Z* are the ECEF coordinates. The scale factor of *(1 + μ)* tends towards 1 due to the low scale μ. The rotation matrix *R(Ω$_x$ Ω$_y$ Ω$_z$)* is defined as a sequence of Euler rotations around the X, Y, Z axes. The translation vector *(t$_x$, t$_y$, t$_z$)* gives the coordinates of the origin of the ECEF coordinate system with respect to ENU coordinate system [Marel, 2014]. In order to construct the ENU system, the ECEF coordinates of the reference points should be rotated and translated from the geocenter to the oblique origin [Wang et al., 2013]:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} -sin\lambda_0 & cos\lambda_0 & 0 \\ -sin\varphi_0\, cos\lambda_0 & -sin\varphi_0\, sin\lambda_0 & cos\varphi_0 \\ cos\varphi_0\, cos\lambda_0 & cos\varphi_0\, sin\lambda_0 & sin\varphi_0 \end{bmatrix} \cdot \begin{bmatrix} X_P - X_0 \\ Y_P - Y_0 \\ Z_P - Z_0 \end{bmatrix} \tag{5.5}$$

where X_P, Y_P, Z_P are the ECEF coordinates of the reference points, X_0, Y_0, Z_0 are the ECEF coordinates of the centroid of the 3D model, which is the origin of the ENU coordinate system. The 3x3 rotation matrix includes the latitude φ_0 and longitude λ_0 of the centroid point.

In order to position the BIM model correctly on the base map, it is necessary to link the real-world reference points to their corresponding points in the 3D model and apply the transformation matrix from Eq. 5.5 to these points. In Three.js, the data associated with the vertices of the 3D model are stored in the **geometry** property, which contains the vertex positions, face indices, normals, colors, and custom attributes. The attributes of the vertices of each element in the BIM model are stored in buffers and their positions are structured in an array [Three.js, 2018]. It is relatively easy to add new vertices to the array by specifying their X, Y, Z coordinates. However, it is difficult to manipulate the existing array due to large amount of vertices stored in it. Thus, it is not possible to find the exact location of the reference points in the 3D model and transform them by using the rotation and translation matrices.

The coordinate conversion and datum transformation are done by implying the aforementioned formulas in Three.js. First, the latitude and longitude of the reference points are stored in a JavaScript array and converted to ECEF in a predefined function **convertToECEF** based on Eq. 5.2 (see Algorithm 10).

Algorithm 10 Convert coordinates with Three.js

coordinatesWGS ← array of coordinates of reference points obtained from Google Maps
coordinatesECEF0 ← ECEF coordinates of reference point 1
coordinatesECEF1 ← ECEF coordinates of reference point 2
coordinatesECEF2 ← ECEF coordinates of reference point 3

function CONVERTROECEF(φ, λ, h)
  a ← 6378137
  f ← 1 ÷ 298.257223563
  e ← $2 \times f - f^2$ ▷ squared eccentricity
  N ← $a \div \sqrt{1 - e \times sin^2\varphi}$
  X ← $(N + 0) \times cos\varphi \times cos\lambda$
  Y ← $(N + 0) \times cos\lambda \times sin\varphi$

$Z \leftarrow ((1-e) \times N + 0) \times sin\varphi$

After obtaining the ECEF coordinates, the local reference system is constructed. For that purpose, the rotation and translation matrices are created as 4x4 matrices, as these are the most commonly used matrices for transformations in 3D computer graphics [Three.js, 2018]. In Three.js, it is not possible to transform a particular point within a 3D object, which is required for aligning the BIM model with the base map. Thus, the rotation matrix is applied to the original matrix of the objects in the 3D model (see Algorithm 11). In addition, the objects coordinates are rearranged, so that the Z coordinate is in the up direction. By default, the Y coordinate points up in Three.js similar to the BIM design platforms.

Algorithm 11 Construct transformation matrix and apply it to the 3D model

function LOADFILE(scene, info)
  function LOAD(object)
    ...
    $\varphi \leftarrow$ latitude of bridge centroid
    $\lambda \leftarrow$ longitude of bridge centroid
    shift $\leftarrow$ translation vector ▷ subtracted ECEF coordinates of two reference points
    $\text{rotationMatrix} = \begin{bmatrix} -\sin\lambda & -\cos\lambda & 0 & 0 \\ -\sin\varphi\cos\lambda & -\sin\varphi\sin\lambda & \cos\varphi & 0 \\ \cos\varphi\cos\lambda & \cos\varphi\sin\lambda & \sin\varphi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
    $\text{translationMatrix} = \begin{bmatrix} 1 & 0 & 0 & shift.x \\ 0 & 1 & 0 & shift.y \\ 0 & 0 & 1 & shift.z \\ 0 & 0 & 0 & 1 \end{bmatrix}$
    transformation matrix = rotationMatrix $\times$ translationMatrix
    apply rotation matrix to object
    set Z coordinate of object in up direction
    ...

The proposed method for georeferencing the BIM model did not give the expected outcome, as the 3D object was not transformed properly. This can be explained by the fact that the reference points could not be identified in the 3D model and the transformation matrix could not be applied directly to the points of interest. This resulted in incorrect rotation of the elements in the model (see Figure 5.12).

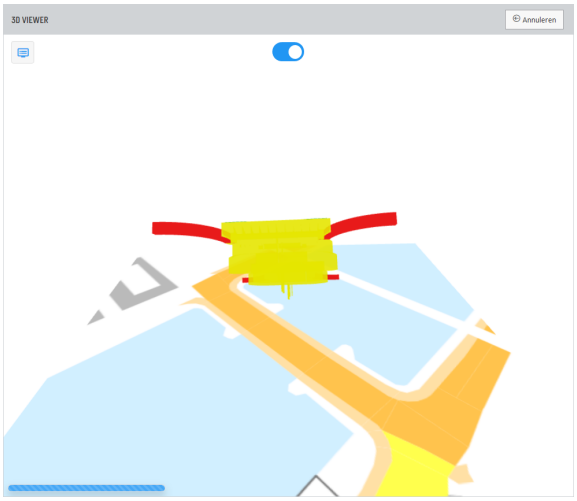
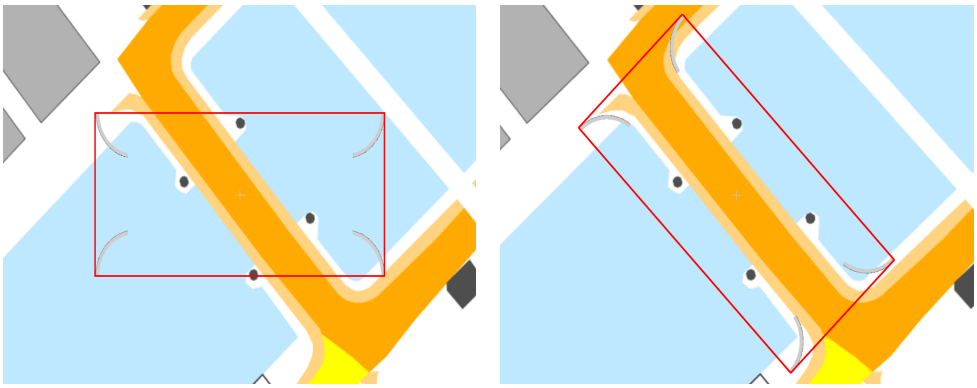


Figure 5.12: BIM model transformation.

Due to the wrong positioning of the BIM model, an alternative approach for transformation is used in CloudCompare. CloudCompare is a software for mesh and point cloud

processing, which provides a transformation tool that allows the user to move and rotate selected objects relatively to other objects in the scene [CloudCompare, 2015]. The advantage of this tool is that a 4x4 transformation matrix is automatically generated and can be directly applied to the 3D model in Three.js.

First, the image obtained from the WMS request and the OBJ file of the retaining walls are loaded in CloudCompare. This OBJ file is chosen, because the retaining structure has the largest spatial extent from all elements of the BIM model. It is also possible to load all OBJ files, but the transformation procedure needs to be repeated for each file. After the required files are loaded, the units of measurement of the OBJ file are converted from inches to meters, as the topographic map in the image is in the Dutch CRS. In the next step, the position of the BIM model is modified to fit in the image. In order to achieve that, the translation in X and Y directions is defined based on the center of the image, which is specified in the image parameters in CloudCompare. Afterwards, the rotation around the Z axis is done manually by approximating the location of the 3D model in the area of interest.



(a) (b)

Figure 5.13: Transformation of the BIM model in CloudCompare: (a) Translation of the model to fit in the image. (b) Manual rotation of 3D model to align with the base layer.

The 4x4 matrix generated in CloudCompare is applied to the objects in the BIM model instead of the transformation matrix from the previous georeferencing approach. Then, the BIM is located at the center of the base layer by setting up the X and Y position (see Algorithm 12). As the units of measurement are changed to meters after the transformation in CloudCompare, the position of the camera is adjusted so that the 3D model does not appear too small in the scene.

Algorithm 12 Apply transformation matrix generated by CloudCompare to the 3D model

```
change camera position
function LOADFILE( scene, info )
    function LOAD( object )
        ...
        transformationMatrix = [ 0.016862   0.018996   0.000000   93811.90625
                                -0.018996   0.016862   0.000000   436408.6875
                                 0.000000   0.000000   0.0254     0.000000
                                 0          0          0          1          ]
        apply transformation matrix to object
        set X and Y position of object to 0
        ...
```

It is evident that the transformation matrix generated by CloudCompare and applied to the 3D model fixed the rotation and translation of the BIM model so that it can align with the base map (see Figure 5.14).

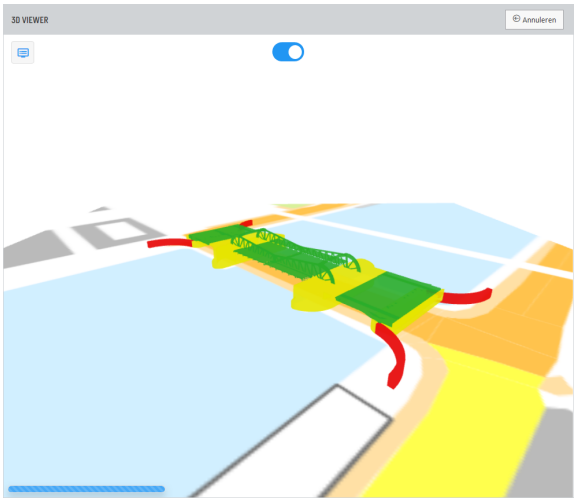


Figure 5.14: BIM model transformation obtained from CloudCompare.

5.2.4 Object Selection

The visualization of the BIM model and integration of the 3D viewer in OBSURV are fundamental functionalities of the prototype. Nevertheless, the ability to select elements of the model and show information about them is also of great importance for the stakeholders. This is achieved by using the **mousedown** event, which is executed when the user presses the mouse button over an element in the scene. This JavaScript event is added to the canvas to allow for selecting objects only within the WebGL renderer (see Algorithm 3).

Three.js provides a specific function for mouse picking, which uses ray casting to find intersections with the elements added to the scene (see Figure 5.15). The ray is created with origin at the mouse position and camera direction. The mouse position is calculated in normalised device coordinates, where the X and Y coordinates should be between -1 and 1 and the Z coordinate refers to the ray direction. Then, these coordinates are added to a 3D vector, which is unprojected with the camera's projection matrix [Three.js, 2018].

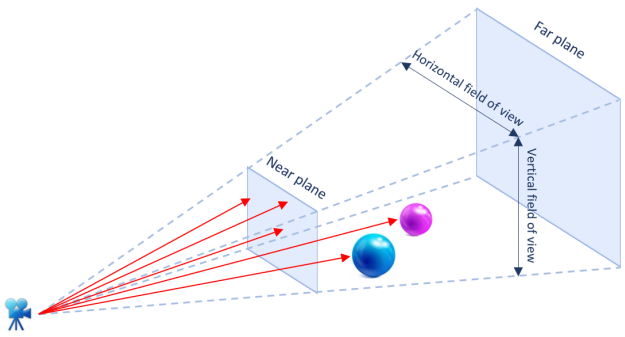


Figure 5.15: Ray casting method implemented in Three.js, adapted from Dirksen [2013].

After setting up the mouse coordinates and ray caster, the object highlight is configured. It is important for the object picking that only the currently selected element of the BIM model is highlighted. This is achieved by assigning a different colour to the element intersected with the ray for better recognition and obtaining the original material of the element in case of no intersection (see Algorithm 13). This procedure needs to be repeated for each inspection year controlled in the history tab in order to update the object selection and highlighting in the 3D viewer.

A separate information tab is created to show additional information about the selected object. The tab includes HTML input fields, which are filled in with the inspection data stored in the userData property of the selected element (see Algorithm 13). In addition, a timeline is integrated in the tab to give a better overview of the condition of the 3D object within the time period specified in the history tab. The time line is represented as a line with rectangles on top of it, which change their colours for each element in the BIM model. The colours of the distinct scores are defined in a dictionary and the CSS styling of the rectangles is updated based on the scores of the selected object (see Algorithm 13). The information tab is made visible only when an object is selected.

Algorithm 13 Configure object selection and highlight functionality

```
function ONMOUSEDOWN( event )
    mouse ← specify mouse position on the screen
    raycaster ← configure ray caster according to mouse and camera
    intersects ← get the closest point to intersection
    if intersection then
        for point in intersections do
            make information tab visible
            get current colour of selected object
            assign blue highlight colour to selected object
            ▷ populate text fields in information tab
            add type of object
            if object ← span locks then
                add electromechanical part as type of element
            else
                add civil engineering part as type of element
            add element name, inspection date, inspector name
            call showCondition for timeline
    else
        make information tab invisible
        remove highlight colour and assign initial colour to the object

function SHOWCONDITION( score1, score2, score3 )
    colours ← dictionary with a defined colour for each score
    add background colour to the squares on timeline
    add background colour for second square on timeline
    add background colour for third square on timeline
```

Figure 5.16 below shows the object selection and highlight created for each bridge element in the BIM model and the information tab with the inspection data on the upper right corner of the 3D viewer. During the implementation of the object selection functionality it became apparent that the entire bridge substructure could not be selected because of the split tool used in the BIM model decomposition (see Section 4.2). A substantial problem with the object selection is the mouse picking function, which is

related to the fact that traditional Three.js applications are standalone and take up the entire HTML document for rendering. Initially, the mouse coordinates were set up with regard to the HTML document, which resulted in wrong position of the mouse. In that case, it is necessary to click outside the boundaries of the 3D model to approximate the document coordinates and select the object of interest (see Figure 5.16).

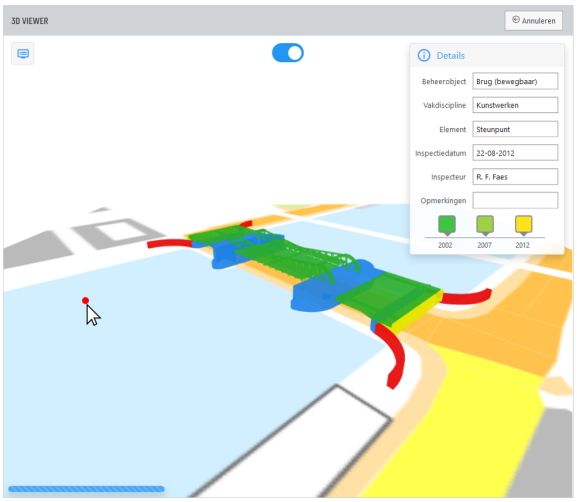


Figure 5.16: Object selection and highlight functionality.

The solution of this issue is to obtain the size of the collapsible APEX page and define the mouse position with respect to the that page. The boundaries of the WebGL canvas, where the 3D model is rendered, are retrieved by using the **getBoundingClientRect** method that returns window coordinates for the element calculated from the window left-upper corner [Kantor, 2018]. The **clientX** and **clientY** refer to the X and Y coordinates of the mouse pointer corresponding to the entire HTML document [Mozilla, 2018a]. It is required to subtract the left and top canvas offset to obtain the object coordinates within the collapsible page (see Figure 5.17).

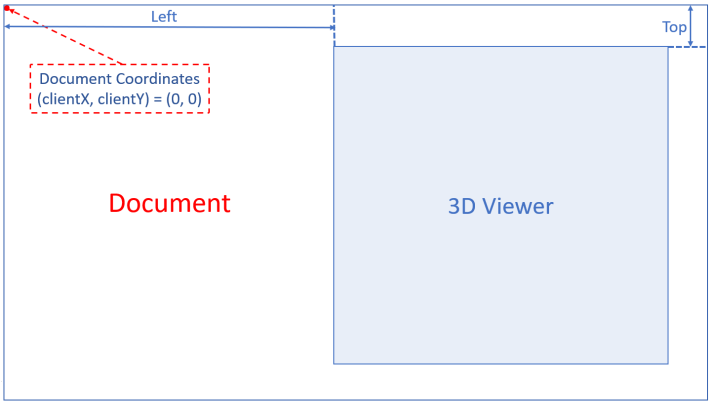


Figure 5.17: Specification of the mouse position in the HTML document and the window coordinates in the 3D viewer, adapted from Kantor [2018].

The refinement of the mouse picking function by determining the boundaries of the collapsible page solved the problem with the mouse position, i.e. the user is able to the click select the object of interest and the same object is highlighted in a blue colour (see Figure 5.18).

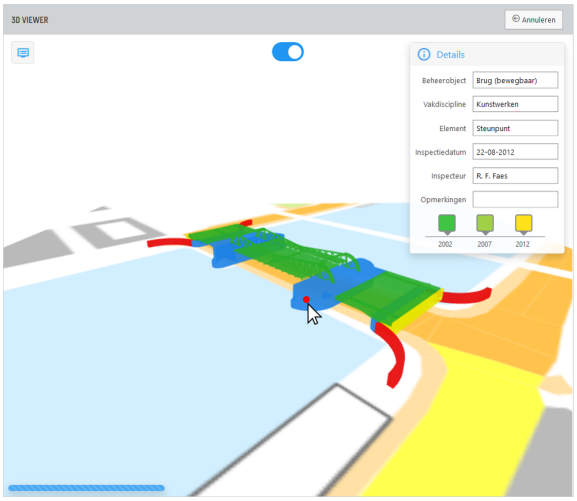


Figure 5.18: Object selection and highlight functionality with fixed mouse position.

5.2.5 Additional Functionalities

A switcher is configured to allow the user to control the objects appearance in the 3D viewer. The labels of the HTML check boxes are sorted by the bridge elements loaded previously. To enable the switcher, the **showObject** function is executed when a check box is clicked. The function takes a list of the respective check boxes as input and verifies if a check box is clicked or not. Then, the corresponding 3D object is made visible or invisible based on the click event (see Algorithm 14).

Algorithm 14 Configure switcher functionality

```
function SHOWOBJECT
    for each check box in the switcher tab do
        if check box is checked then
            make the corresponding object visible
        else
            make the corresponding object invisible
```

The switcher can be combined with the animation functions to make small bridge elements more visible, e.g. the span locks that are hardly recognisable in the normal view of the model. The rotation and zoom functions enable the user to see some of the object components due to the colour transparency of the bridge elements (see Figure 5.19).

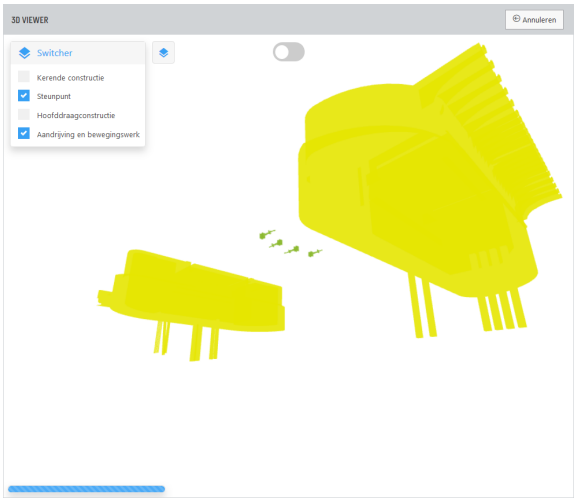


Figure 5.19: Bridge substructure and span locks in combination with the rotate and zoom functions.

The change detection functionality is represented in the form of radio buttons so that only one item is selectable at a time. The radio buttons are labelled with the inspection years, which allows the user to choose the year of interest and see the changes in the condition of the distinct elements. When the user selects the year, the **showHistory** function is executed to assign the corresponding colours to the 3D model. Similar to the initial colour assignment, each object in the BIM model is traversed and the material of its components is changed automatically (see Algorithm 15).

Algorithm 15 Configure change detection functionality

```
function SHOWHISTORY
    for each object in the objects array do
        traverse object components
        if year 2007 selected then
            colorComponent( component, object score for 2007 )
        else if year 2002 then
            colorComponent( component, object score for 2002 )
        else
            colorComponent( component, object score for 2012 )
```

The tab showing the inspection results from previous years contains the years of the last three inspections of the infrastructure asset (see Figure 5.20). The time changes are also visible in the information tab displayed when a bridge element is selected in the 3D viewer (see Figure 5.18). Ultimately, the user should be able to enter the results of the NEN inspections directly in the viewer and a new radio button with the next inspection year should appear in the change detection tab.

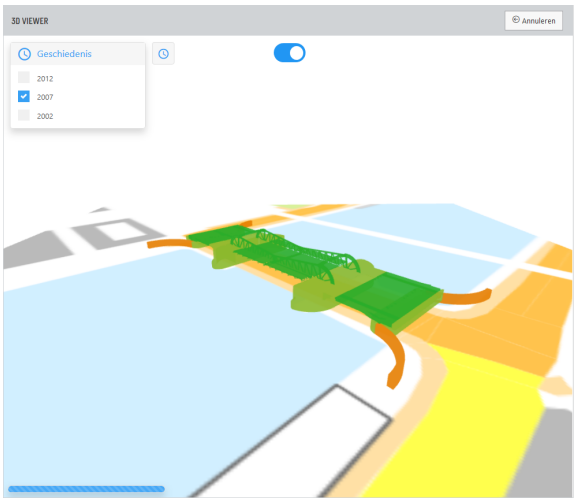


Figure 5.20: Change detection functionality in the 3D viewer.

In order to make the 3D viewer responsive and resizable on different devices, the size and viewport of the WebGL renderer should be changed according to the width and height of the collapsible page. Furthermore, the BIM model should be scaled according to the size of the collapsible page. For that purpose, the camera aspect and projection should be updated to allow for the correct rendering of the 3D model in the scene (see Algorithm 16).

Algorithm 16 Make WebGL renderer resizable

```
function ONCANVASRESIZE
    change renderer size to the canvas size
    change renderer view port to the canvas aspect
    camera aspect = canvas width ÷ canvas height
    update camera projection matrix
```

As the 3D viewer is embedded in the existing application, it is dependent on other HTML elements and some settings in the APEX page needed to be adjusted to resize the WebGL renderer properly (see Figure 5.21).

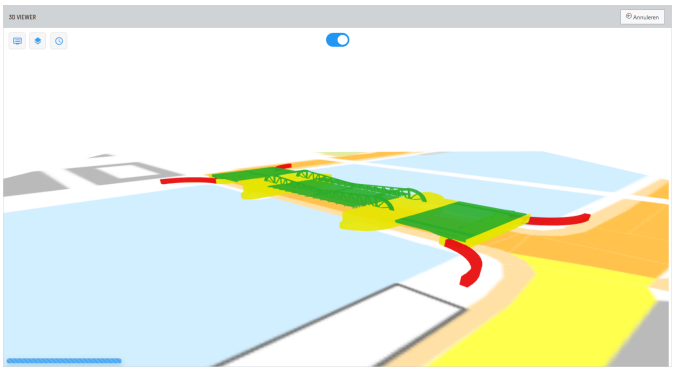


Figure 5.21: Resized WebGL renderer and 3D model.

6 PROTOTYPE EVALUATION

After the research methodology regarding the development and implementation of the 3D prototype has been explained, the focus is placed on the evaluation of its usability and functional operability. This chapter presents the concept of usability testing, which is the most suitable approach to assess the user interaction with the prototype, and describes the evaluation process along with the analysis of the results from the conducted survey.

6.1 USABILITY TESTING

Usability testing can be described as a research tool, which enables the users to assess a product or a prototype. The extent of the usability tests can range from classical experiments with large sample sizes to informal qualitative studies with only one participant. For private organisations, usability testing attempts to eliminate design problems and frustration for the users in order to release useful, effective, efficient, and satisfying products. Furthermore, the product evaluation aims to improve the profitability of products by keeping track of test results and creating a historical record of usability benchmarks for future releases [Rubin and Chisnell, 2008].

The usability testing has a great value for developers, as it provides essential information about the quality and usability of their applications, and helps them recognise any possible issues found by the users [Ivory, 2003]. Therefore, the approach is of great importance for the assessment of the prototype and its further implementation by means of identifying problems related to its development and operation, and indicating potential improvements. Nevertheless, the method has some limitations related to its reliability and choice of participants [Rubin and Chisnell, 2008]:

- **Testing is an artificial situation:** Usability testing depicts an actual situation of usage, but not the situation itself, which in turns can affect the results of the study.
- **Test results do not prove that a product works:** The results of usability testing are highly dependent on the way the test is conducted and they cannot be fully relied on.
- **Test participants are rarely fully representative of the target group:** Usability testing usually encompasses a large group of participants, although the actual end users are often hard to identify and can differentiate from the target group.
- **Testing is not always the best technique to use:** Alternative techniques can be used for the assessment of products and prototypes to reduce cost and time, and ensure more accurate results. For instance, an expert review enables a prompt identification of usability issues through the examination of a UX specialist.

The expert evaluation focuses mainly on the user friendliness of applications by assessing the UI design and human-computer interaction [Harley, 2018]. In order to

derive indepth conclusions about the usability and functional capability of the 3D prototype, it is desirable to do testing with users. Usability testing usually begins after the creation of a paper-based UI design and continues through the prototype development. However, most web sites are not evaluated before their implementation in reality [Lazar, 2001].

Each testing approach has different objectives and can be performed in different ways [Rubin and Chisnell, 2008]. In general, there are two types of data that can be collected in a usability test: qualitative and quantitative data. The qualitative study is based on observational findings by UX experts to directly assess the ease of use of a system and determine whether the UI features are sufficiently designed. The quantitative research reflects the user performance on a given task or the participants' perception of usability [Budiu, 2017]. Both the qualitative and quantitative studies use multiple research techniques, such as surveys and interviews, to gather and analyse empirical data [Neuman, 2014].

6.2 SURVEY RESEARCH

The purpose of the 3D prototype evaluation is to assess the usability of the prototype and provide optimisation measures for its further implementation. The quantitative research is suitable for assessing the functional operability of a system rather than identifying the main design problems [Budiu, 2017]. Therefore, this approach is chosen for evaluating the usability of the 3D viewer.

Survey research is a quantitative method for collecting data from a large number of people to gather information on their beliefs, opinions, or expectations in specific situations. Surveys can provide accurate, reliable, and valid data that can be analysed with statistics and presented in charts, graphs, or tables. Therefore, this type of quantitative research is most frequently used in descriptive and explanatory research [Neuman, 2014]. The survey research follows a specific sequence of steps, which are represented in Figure 6.1 below.

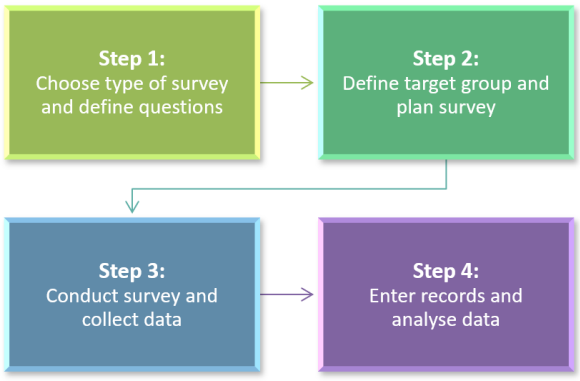


Figure 6.1: Steps in the survey research process, adapted from Neuman [2014].

Step 1: Choose Type of Survey and Define Questions

The first step of the research is to create an instrument. The survey can be conducted in the form of a survey questionnaire, where the participants read and answer the questions themselves, or an interview schedule, in which the interviewer asks the respondents a set of questions and records their answers [Neuman, 2014]. In case of the 3D prototype evaluation, a questionnaire survey is preferable for easier data collection and analysis.

After choosing the type of survey, it is necessary to define questions and design the form. Questions are fundamental in the surveys and determine which indicators are used for testing. Therefore, the questions should be designed in a way that the users are able to assess the effectiveness and feasibility of the created prototype, and expose any implementation issues they encounter [Cabaj, 2017]. When designing the questions, several aspects need to be considered for valid and reliable results. Possible confusion should be avoided and the respondents' perspective should be kept in mind to ensure that the participants understand the meaning of each question as intended and are able to answer completely [Neuman, 2014].

Survey questions are usually classified into closed and open-ended questions. Closed questions enable the participants to choose the response from a fixed set of answers, which makes the questions easy to answer and analyse. However, the respondents are not able to express a more complex meaning and the survey can be biased, if the participants tend to systematically tick either the first or last answer. Open-ended questions allow the respondents to freely express their opinions. In this way, additional information about the topic can be gathered in case there is not enough existing information available. Unlike closed questions, the open-ended questions require more time to be answered and are more difficult to analyse due to their diversity. Thus, it is necessary to create a system to classify the responses [Siniscalco and Auriat, 2005].

Step 2: Define Target Group and Plan Survey

Before conducting the survey, it is important to specify the target group of participants based on the aim of your application. The end users of the 3D prototype are asset managers in the Dutch municipalities, who are responsible for the inspections and maintenance of the civil structures. However, no asset managers from a public institution could be involved in the evaluation due to lack of communication. Not enough asset managers were found or agreed to complete the survey and a small amount of users does not guarantee reliable results. In this case, the target audience is comprised of several specialists in different AM domains and BIM development at Sweco, which are able to critically evaluate the functional capability of the 3D viewer and suggest further improvements.

In the next step, the location of the survey should be defined, it can be done in person, via mail or telephone, or over the Internet by allowing the participants to fill in a form [Neuman, 2014]. Due to the diversity of respondents in the questionnaire, it is more convenient to do an online survey.

The questionnaire is created in Google Forms, a well-known tool for creating and analysing surveys. The form is divided into 5 different sections, including welcoming and instructions, participants' personal details, usability testing, suggestions for further improvements and additional comments, and closing (see Appendix F). It is rather challenging to design clear and relevant questions for respondents with diverse backgrounds, as the question wording may be interpreted differently by everyone, but all answers should be combined for the data analysis [Neuman, 2014]. Therefore, simple and straightforward closed and open-ended questions are developed for the questionnaire.

The online survey is sent to the participants in the 3D prototype evaluation after the project has been presented to the audience along with a live demo showing all functionalities of the created 3D viewer.

Step 3: Conduct Survey and Collect Data

After choosing the target group and planning the questionnaire survey, some instructions on completing the survey should be given to the participants [Neuman, 2014]. In case of the 3D prototype evaluation, the instructions are included in the beginning of the questionnaire.

For the data collection, it is important to keep track of the respondents' answers to every question in the questionnaire to enable easier data storage and analysis [Neuman, 2014]. All the responses from the participants are stored in Google Forms and are visible for the creator of the survey. In addition, the responses can be saved as spreadsheet for further investigation.

Step 4: Enter Records and Analyse Data

Once the survey data are collected, the responses from the individual questionnaires are reviewed and stored in a computer-readable format for comparisons and statistical analysis. There are various statistical methods that can be used in the analysis of survey data, such as descriptive and inferential statistics. The results from the survey data analysis can be presented in graphs like bar charts, pie charts, histograms, etc [Neuman, 2014]. In order to ease the analysis process, Google Forms provides a summary of the answers by question in the form of text and graphs depending on the type of question.

The survey was completed by 9 respondents. The personal details section provides information on the background of the participants and assesses their familiarity with OBSURV.

Occupation

All the participants come from different areas of occupation. Two of them are advisers for AM for civil structures. The rest of the target group are specialists in the fields of waterway construction, civil engineering, 3D development, business development, and consultancy (see Table 6.1). Additionally, the product manager of OBSURV completed the survey as well.

Table 6.1: Occupation field of the participants in the survey.

Occupation Fields	Responses
Asset Management Civil Structures	2
Waterway Construction	1
Infrastructure Design	1
Civil Engineering	1
3D / VR Development	1
Business Development	1
Consultancy	1
Product Management	1

Familiarity with OBSURV

The pie chart in Figure 6.2 illustrates the familiarity of the respondents with OBSURV expressed in percentage terms. It is clear from the graph that all the participants know the web application. The majority of them are very familiar with it, as they presumably use it on a daily basis. Nearly a third of the respondents are acquainted with OBSURV, but have never used it before. Around 11% said that they used the Web GIS several times.

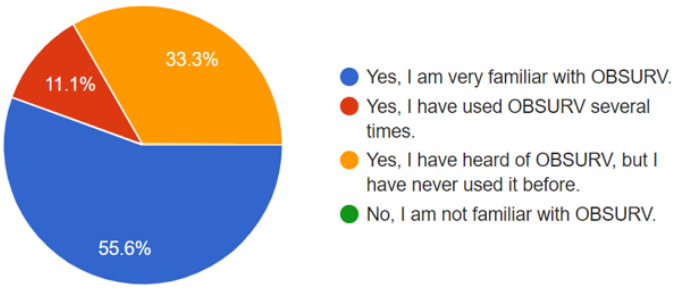


Figure 6.2: Pie chart showing the familiarity of the participants with OBSURV.

Navigation around the 3D viewer

The usability testing in the next questionnaire section aims to assess the functional operability of the 3D prototype. All the respondents shared the same opinion that navigating around the 3D viewer is easy and intuitive.

Grading of the 3D viewer features

The participants graded the functionalities provided in the 3D prototype very differently. The stacked bar chart presented in Figure 6.3 shows the features in the 3D viewer on the horizontal axis and the number of respondents on the vertical axis. The designated means of grading is a standard 5 tier scale ranging from poor to excellent. The sorting of the features in the graph is based on the overall score consisting of the sum of responses for each grade level multiplied by a weight factor from 1 to 5, in a high to low order.

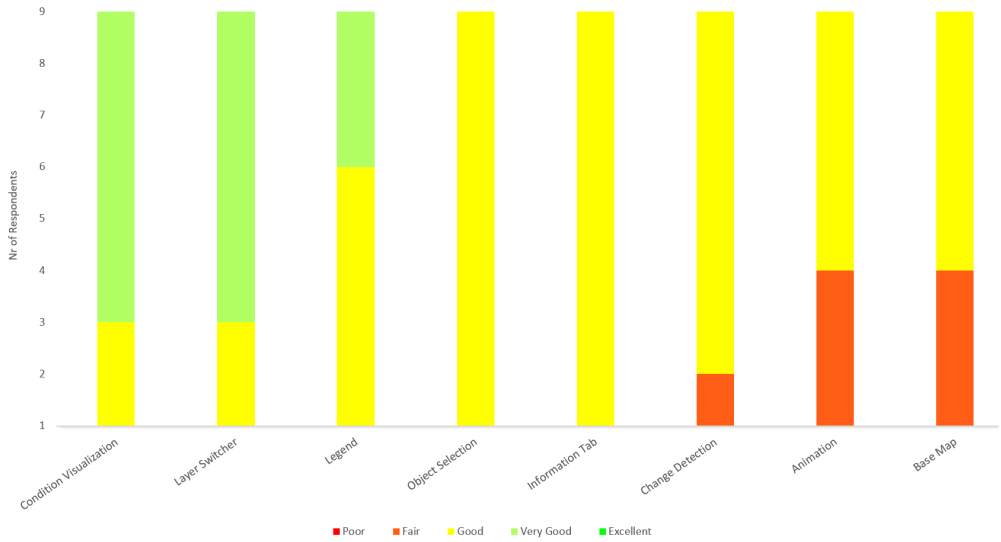


Figure 6.3: Bar chart showing the grades of the 3D viewer functionalities given by the participants.

The functionality with the highest ranking based on the calculated overall score is the condition visualization, receiving 7 grades of very good and 2 grades of excellent. Following in second place is the layer switcher with all the participants grading it as very good. The legend, object selection, information tab, and change detection functionalities received good or very good grades from the respondents. The animation and base map were graded relatively good with the exception of 2 people who assessed the features as fair. The details of the calculated overall score are provided in Table 6.2 below.

Table 6.2: Calculation of the overall score of the functionalities of the 3D prototype.

Feature	Poor	Fair	Good	Very Good	Excellent	Score
	$N \times 1$	$N \times 2$	$N \times 3$	$N \times 4$	$N \times 5$	
Condition Visualization	0	0	3	24	10	**37**
Layer Switcher	0	0	3	32	0	**35**
Legend	0	0	6	28	0	**34**
Object Selection	0	0	9	24	0	**33**
Information Tab	0	0	9	24	0	**33**
Change Detection	0	2	12	12	5	**31**
Animation	0	4	9	12	5	**30**
Base Map	0	4	9	16	0	**29**

Most useful functionality of the 3D prototype

The pie chart in Figure 6.4 shows which functionality implemented in the 3D viewer is the most useful one according to the participants. As expected from the overall scores of the features, visualizing the condition of the distinct bridge elements is highly valued by the vast majority of respondents. The change detection tool and the animation functions are marked as the most useful features by two of the participants.

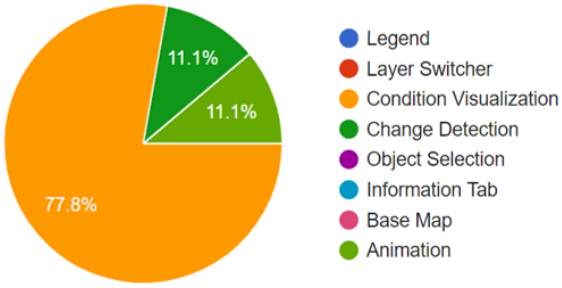


Figure 6.4: Pie chart showing the most useful functionality of the 3D viewer according to the participants.

Main usability issues of the 3D prototype

Afterwards, the respondents gave their opinion on the main usability issue of the 3D prototype. The participants exposed diverse problems related to semantic 3D models and the operability of the 3D viewer:

- Lack of BIM models of infrastructure assets in the public institutions, which makes the collection of 3D data more difficult
- Rendering of multiple 3D objects in OBSURV due to the large size of BIM models, which can affect the performance of the application
- Visualizing the bridge elements, as the majority of NEN inspections are done on the object components level
- Not enough information about the objects and their inspections available, which limits the usability of the 3D prototype

Further improvements

In the end of the 3D prototype evaluation the participants were asked to give suggestions for further improvements, which can be used for the implementation of the prototype in the near future. The proposed enhancements are related to the UX and technical features that should be developed:

- Replacing the condition score with risk analysis to provide more information about the condition of the civil structure and its components
- Adding more electromechanical components of the civil structure to enable better data management of all object components
- Combining maintenance with design data to link data available in distinct LCAM phases
- Switching between maintenance status visualized in different colours and normal view of the initial BIM model in gray colour for concrete, brown for wooden materials, etc
- Using larger infrastructure models of roads and railways to use the BIM models also for the construction and engineering LCAM phases
- Placement in 3D Google Maps for more sophisticated data representation

- Graphical improvements, e.g. making the objects more transparent when another one is selected by the user and adding a time slider for better representation of the change detection of the object elements in the information tab

6.3 DISCUSSION

The BIM-GIS integration is still a relatively new topic in the Geomatics field. One of the main reasons for the deployment of detailed building and construction data in GIS systems is the lack of spatial analysis capabilities in BIM, which can be compensated by the diversity of spatial relationships between topographic objects in GIS. In that sense, the integration can enhance the current practice of data sharing between the tools used in the LCAM processes. Nevertheless, the BIM and GIS systems are incompatible with regard to the modelling environment and reference systems, e.g. the spatial data are georeferenced and the BIM models are created based on a local coordinate system [Karan et al., 2015].

A considerable problem in the research was the collection of BIM models. As the BIM technology has become more popular just recently, the municipalities are still migrating towards 3D design and modelling. Thus, there are not so many BIM models of infrastructure assets available within the public institutions, which makes the data acquisition difficult. It is expected that BIM processes will be fully integrated in the work procedure in the near future and it will be easier to obtain semantically rich 3D data for LCAM.

Due to the variety of BIM modelling platforms, the design techniques in the BIM technology differ significantly. The BIM professionals are usually free to choose the approach how to build BIM models and which software package to use. As a result, it is rather difficult to find consistent 3D data and process it accordingly. Ultimately, the developed 3D prototype can be used for the integration of any BIM models in Web GIS systems like OBSURV. However, the 3D models need to be processed beforehand because of the inconsistent data sources.

The BIM decomposition focuses on the object elements, which limits the functionality of the 3D prototype. As stated in Section 2.1, most NEN 2767-4 inspections are done on object components for more realistic estimation of their condition. Thus, it is advisable to process the BIM model in such a manner that the distinct building parts of an object element can be clearly identified to allow the users to view and edit more detailed data of their assets. Furthermore, the decomposition of the assets on components level will enhance the visualization and usability of the BIM models, and enable the asset managers to manage their data more efficiently.

In the context of the BIM decomposition, the structure of the initial 3D model is another topic for consideration. Not all elements of movable bridges specified in the NEN 2767-4 standards were included in the provided BIM model, e.g. the fence and some electromechanical parts of the bridge were missing. Therefore, only a few elements could be used in the decomposition and visualized in the 3D viewer.

The WebGL framework used for representing the BIM model proved to be suitable to meet the major user preferences. Three.js offers fast and straightforward rendering of interactive 3D models. However, the developed 3D prototype differs significantly from the majority of sample applications, which visualize relatively simple 3D objects with basic functionalities and are not customised with additional HTML and CSS content. Furthermore, the majority of applications based on Three.js are standalone and not embedded within existing applications, which posed challenges to the integration of the 3D viewer in OBSURV.

The limitation of Three.js is the lack of georeferencing tools, for which a suitable solution should be found in case of integration of BIM models in GIS systems. This problem can be solved by using iTowns created for visualizing geospatial data, but it is uncertain if the library handles OBJ files and supports the functionalities of the 3D viewer required by the stakeholders.

Two different methods for georeferencing the BIM model were presented in this research. The first approach uses the geographic coordinates of three reference points of Koninginnebrug to construct a transformation matrix, which modifies the rotation and position of the BIM model. However, the 3D model was not georeferenced properly, because the reference points could not be identified in the BIM model. The other approach utilises a transformation tool provided in CloudCompare, which automatically generates a transformation matrix. For that purpose, a WMS image of the area of interest in Rotterdam is loaded in the software and the 3D model is translated to fit the image. The rotation of the BIM model is done manually, which is error-prone due to the approximation of the exact position of the bridge in the real world. Therefore, further research on the automation of the transformation process is required.

Another point of discussion is the performance of the developed 3D prototype. Performance is of great importance, as it plays a major role for the applications to run smoothly and meet the user preferences. Current web services are characterised by a large number of concurrent requests and huge data transmission, which requires complex computation for satisfactory performance [Tu et al., 2004]. The key measurements in a software performance are the response time and space memory due to their direct impact on the end users [Yang et al., 2011].

It has been estimated that an acceptable waiting time for a web response is about 8 seconds [Yang et al., 2011]. This time frame is very limited for the visualization of BIM models in Web GIS. Although the 3D model of Koninginnebrug is not as large as other more complex BIM models, the average time it takes to load all bridge elements is 20 seconds. This can be associated with the internet bandwidth - a higher bandwidth can increase the number of requests handled per second [Loechel and Schmid, 2012]. Thus, it is advisable to use broadband internet connection, which is stronger than the wireless connection.

The slow web environment can also be linked to memory problems. The memory is a complex function of the server, network, and client, which can cause multifaceted problems in the application performance [Yang et al., 2011]. In case of the integration of a single BIM model, no memory issues were indicated. However, the 3D prototype will eventually be implemented in the next few years and more 3D models will be

incorporated. Therefore, a better data storage strategy is required. Currently, all OBJ files are stored on the server, but it is also possible to store them in the Oracle database by using CLOB or BLOB data types for handling large objects.

7 CONCLUSIONS AND FUTURE WORK

This chapter summarises the findings of the conducted study and provides suggestions for future research. The objective of this project is to study the integration of BIM models in a Web GIS for public space management by solving issues related to the collection, processing, and web visualization of semantic 3D models. As a result, a 3D prototype is developed to improve the current data representation and support dynamic decision-making processes in the maintenance phase of LCAM.

7.1 CONCLUSIONS

The research covers a wide range of processes to achieve the desired goal and seeks to answer the research questions presented in Section 1.2. Below there is a summary of the answers to these questions. More detailed information about the procedures is provided in Sections 2.1, 3.1, 4.2, and 5.2.

1. *What is the current state of data visualization in LCAM and how can it be improved?*

   Currently, the systems in LCAM are based on 2D data, which are represented in the form of tables, reports, and thematic maps. This enables the asset managers to perform basic tasks like adding and deleting objects, and updating their attributes. However, it limits the visualization of complex infrastructure assets, such as civil structures. Therefore, semantically rich models can improve the management of these assets by providing detailed and more realistic representations that are easy to interpret by diverse stakeholders. BIM is a widespread technology for semantic 3D modelling, which has already been incorporated in the planning and construction LCAM phases to expand and refine the design techniques. These models can be preserved and processed for the asset maintenance and operation to allow for a more integrated workflow in LCAM.

2. *What are the requirements for the development of the 3D prototype?*

   It is important to define the services that the system must provide to fulfil the user preferences in order to determine the objective of the prototype. The requirements for the design and functional capabilities of the 3D prototype were specified by different stakeholders involved in the project.

   BIM models were chosen as data source for integration in the Web GIS system due to their high level of detail and accuracy, as well as ability to manage the entire asset life cycle. In view of the research scope, it is necessary to acquire a BIM model of a bridge in Rotterdam and process it accordingly to comply with the NEN 2767-4 standards for inspection and maintenance. For easier distinction from the 2D view, it is essential to visualize the 3D model in a 3D viewer integrated in OBSURV and link it to its corresponding 2D object.

In addition, it is advisable to assign different colours to the distinct bridge elements based on the inspection data to represent their current condition. Adding a legend to illustrate the meaning of these colours is desirable for a better understanding of the condition scores.

It is important to allow the user to be able to select the elements in the BIM model and see relevant information about them in the 3D viewer to make the 3D prototype more user-friendly.

A base map can be integrated in the 3D viewer to enhance the visualization of the assets in the context of their surrounding environment. For that purpose, the BIM model needs to be georeferenced to align with the base layer. This can be achieved by using a transformation matrix that rotates and translates the 3D model to the right position on the map.

In terms of interactive animation, the rotation, zooming and panning are standard functionalities to implement for a better navigation of the view port. These can be expanded with an explode function to separate the model into pieces and improve the visibility of the distinct parts.

Furthermore, a switcher can be incorporated in the UI to allow the users to control the visibility of the bridge elements. In order to track the changes in the condition of the bridge elements, it is desirable to show the results of previous inspections in the 3D viewer by adjusting the BIM model appearance.

To achieve all the prerequisites, it is necessary to find a suitable technology for development of the 3D prototype. WebGL is used for the rendering of the BIM model, as it can be easily integrated in existing web applications due to its compatibility with all modern web browsers. At the end of the development process, the prototype should be tested by several users or experts in the AM and BIM field to assess its usability and determine whether it meets the predefined requirements.

3. *How can 3D BIM models be processed for effective utilisation in condition monitoring?*

Due to the variety of design techniques in the BIM technology, the composition of the 3D models does not always follow specific standards. In order to be utilised for condition monitoring, the BIM models should be processed according to the national standards for inspection and maintenance. In the Netherlands, the NEN 2767-4 standards are used to inspect different infrastructure assets like roads, green areas, bridges, tunnels, ports, sewage systems, etc. The standards define three separate levels for the composition of each asset: object, object elements, and object components. Thus, the provided BIM model needed to be decomposed accordingly to correspond to this structure. In this research, the 3D model decomposition was performed in the same BIM design platform, which was used for the model creation - Autodesk Inventor. The platform provides simplification and grouping tools to combine or split some parts and obtain the correct elements, as specified in the NEN standards.

4. *How can 3D BIM models be integrated in a Web GIS?*

The 3D prototype was developed based on a WebGL for easier access without the use of any plug-ins. Three.js is a JavaScript library for rendering interactive 3D

graphics and animations, which allows the visualization of complex BIM models and easy customisation of the HTML and CSS context.

The BIM model was integrated in a collapsible page in the Web GIS for clearer distinction between the 2D and 3D view. The page was linked to a table with the records of the public infrastructure assets to make the 3D viewer visible when an object in the table is selected. In order to render the BIM model in the 3D viewer, it was necessary to find a suitable format supported by Three.js. Wavefront OBJ was chosen due to its extensive use in web rendering and additional functions implemented in Three.js.

For representing the current condition of the bridge elements, the objects in the 3D model were coloured according to the scores asigned to them during the NEN 2767-4 inspections and a legend was added to the 3D viewer to show the score ranking.

An information tab was created to show relevant information about the selected object extracted from the database and to visualize the changes in the condition in the form of a time line. The tab was made visible only, when a particular element is selected by the user. The functionality allows the asset managers to view the inspection results in a more user-friendly and interactive way than the tabular representation.

There are several possibilities to integrate a base map in the 3D viewer. A simple 2D overview map can be added to the WebGL renderer as a background by using web mapping tools like Leaflet, OpenLayers, and Mapbox. However, a large map can affect the performance of the web application and does not enable the visualization of subsurface objects. Therefore, a ground plane was added below the BIM model and an image of the surrounding area was attached to the plane. The image was obtained by using the WMS service to request a topographic map from the PDOK server, the Dutch central facility for geospatial data. In addition, a toggle button can be added to allow the users to switch the base layer on and off. In order to align the BIM model with the ground plane, it was necessary to georeference the 3D model by applying a transformation matrix to it.

Two different approaches for georeferencing were presented in this research. The first method uses the geographic coordinates of three reference points of the bridge to construct a rotation and translation matrix. Nevertheless, the 3D model was not georeferenced properly, because the reference points could not be identified in the BIM model. The other approach utilises a transformation matrix generated by CloudCompare to determine the correct position and rotation of the 3D model. The translation of the BIM model is done by fitting the model in the WMS image of the area of interest and the rotation was performed manually. The latter approach successfully located the 3D model on the ground plane.

Three.js has a built-in functionality for the basic animation functions like rotation, zooming and panning that uses orbit controls to allow the camera to move around a target in the scene.

A switcher was incorporated to control the appearance of the layers in the 3D viewer and allow the user to decide which elements should be investigated. The switcher can be represented as a set of check boxes for easier reference to the 3D objects.

A change detection functionality was implemented to show the conditions of the bridge elements over time and udpdate the colours of the BIM model based on the chosen inspection year.

7.2 CONTRIBUTION TO GEOMATICS

Geomatics is an applied science that involves the acquisition, modelling, analysis, management, and visualization of geographic information with the aim to better understand built and natural environments, and solve real-world problems.

This research addresses a popular topic in the Geomatics field and presents a possible solution to overcome the issues associated with the integration of BIM models in GIS systems. A sample 3D interface is created for the visualization of BIM models for condition monitoring based on the requirements of different experts in the AM field. In addition, the study presents a method for the decomposition of BIM models by using a well-known platform for BIM modelling, Autodesk Inventor. Furthermore, the research focuses on the use of the Three.js as one of the most popular WebGL technologies for rendering complex 3D models. As Three.js is not intended for visualization of geospatial data, a solution for the integration of a base map and georeferencing of BIM models is provided.

The majority of researches related to the BIM-GIS integration investigate the representation of 3D models of buildings, which is significantly different from this project that focuses on the management and visualization of distinct public infrastructure assets to improve the decision making in LCAM.

7.3 FUTURE WORK

Due to the limited time of this project, not all the requirements regarding the functionalities of the 3D prototype are met. Therefore, several suggestions for future research are presented below.

In terms of data selection, the functionality to add or update the records of the NEN 2767-4 inspections in the information tab in the 3D viewer needs to be implemented. In order to make the 3D viewer more user-friendly and interactive, it is important to allow the users to enter or modify the inspection data directly in the viewer and see the results immediately in the BIM model. Furthermore, a solution needs to be found for the explode function, which is crucial for a sophisticated visualization of the distinct elements in the BIM models, but not implemented in the prototype due to time constraints.

The 3D prototype was developed mainly for bridges due to their structure complexity. It is necessary to investigate which other public infrastructure assets are suitable for 3D visualization and which are the most appropriate sources of 3D geoinformation for them. It is expected that benches, lampposts, roads, green areas and playgrounds can also be represented in 3D for better data management. BIM models are beneficial for the visualization of civil structures and LiDAR point clouds can represent large scale

areas for a better overview of their surroundings.

The proposed decomposition approach focuses on the grouping of distinct components to obtain new elements and does not change their geometry. Therefore, all decomposed elements are provided in the same LOD. In case more BIM models are integrated in OBSURV, the 3D models need to be generalised by decreasing the LOD in order to reduce the file size. Then, it is more useful to represent the elements in different LODs depending on their importance for the infrastructure assets. For instance, smaller elements like the span locks are essential for the opening and closing of movable bridges and more vulnerable to damages than other bridge elements, and thus, they can be represented in a higher LOD.

A significant gap in the proposed approach for the BIM integration in Web GIS is the lack of connection between the 3D viewer and the design platform where the 3D model has been created., i.e. the changes in the 3D model made in the BIM design platform are not automatically updated in OBSURV. In case some object components are added or deleted, the decomposition of the BIM model needs to be performed manually. This issue needs to be solved, as it causes double work within the integration process.

For the further implementation of the 3D prototype, it is advisable to include risk analysis that provides more detailed information about the condition of the civil structure and its components. This gives a better estimation of the overall asset condition and can be used to make predictions about any possible damages in the future based on the historical inspection data.

In order to link the design and planning phases to the asset maintenance and operation, it is useful to show design information about each element in the 3D model, which is extracted from the initial BIM model or any other data sources. This information contains the structure and material of the elements and gives a better insight for the maintenance planning.

The BIM-GIS integration is a major topic in the field of geospatial technology and construction. Currently, there are no official standards on procedures for integrating BIM models in existing GIS systems, which makes this research and similar ones more difficult. The major challenge is related to the missing geographic context of the BIM models, which can be overcome by georeferencing the models. Therefore, further investigation on the georeferencing techniques is required, as none of the existing methods seems to be truly successful.

In the future, the developed 3D prototype can be expanded by adding AR to improve the visualization and user interaction with the BIM model. The users can benefit from the AR technology, as they are able to view the surface and subsurface elements of the public infrastructure assets in the provided geographic context.

Should the prototype be put into operation and the suggestions from this section are taken into account, then this would significantly improve the condition monitoring of infrastructure assets and ease their maintenance planning.

BIBLIOGRAPHY

Abdul-Rahman, A. and Pilouk, M. (2008). *Spatial Data Modelling for 3D GIS*. Springer Verlag.

Alesheikh, A. A., Helali, H., and Behroz, H. A. (2002). Web GIS: Technologies and Its Applications. In *Proceedings of the Symposium on Geospatial Theory, Processing and Applications*, Ottawa, Canada.

Alexiadi, C. and Potsiou, C. (2012). How the integration of n- dimensional models (BIM) and GIS technology may offer the potential to adopt green building strategies and to achieve low cost constructions . How the integration of n- dimensional models (BIM) and GIS technology may offer th. In *FIG Working Week 2012 Knowing to manage the territory, protect the environment, evaluate the cultural heritage*, Rome, Italy. FIG. http://www.fig.net/resources/proceedings/fig{_}proceedings/fig2012/papers/ts08j/TS08J{_}alexiadi{_}potsiou{_}6061.pdf.

Altmaier, A. and Kolbe, T. (2003). Applications and Solutions for Interoperable 3D Geo-Visualization. In Fritsch, D., editor, *Proceedings of the Photogrammetric Week 2003*, Stuttgart, Germany. Wichmann Verlag. http://www.ifp.uni-stuttgart.de/publications/phowo03/kolbe.pdf.

Autodesk (2012). BIM for Infrastructure: A vehicle for business transformation. Technical report. https://www.my-morpheus.com/UserFiles/file/bim-vehicle-for-business-transformation-whitepaper-en.pdf.

Autodesk (2014). User coordinate system (UCS). https://knowledge.autodesk.com/support/inventor-products/learn-explore/caas/CloudHelp/cloudhelp/2014/ENU/Inventor/files/GUID-FF15BB0D-5FE9-45F9-92FD-D70A3F6A8FD0-htm.html.

Autodesk (2018). Top Products. https://www.autodesk.com/products.

Baumann, P. and Aiordachioaie, A. (2009). Preserving Geo-Scientific Data Assets Through Service Interoperability. In *PV 2009 Ensuring Long-Term Preservation and Adding Value to Scientific and Technical Data*, Villafranca del Castillo, Spain. http://www.alliancepermanentaccess.org/pv/pv2009/PV2009/23_Baumann_GeoDataServiceInterop.pdf.

Biljecki, F., Ledoux, H., and Stoter, J. (2017). *Generating 3D city models without elevation data*, volume 64. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85009347977{&}doi=10.1016{%}2Fj.compenvurbsys.2017.01.001{&}partnerID=40{&}md5=2999c2052780c3cb0d09e45ae14b633b.

BIMForum (2016). Level of Development Specification 2016. Technical report. ftp://ftp.varinex.hu/CGS/LOD{_}Spec{_}2016{_}Part{_}I{_}2016-10-19.pdf.

Boanca, T. (2014). *BIM – GIS integration for Asset Management*. Master's thesis, Wageningen University and Research Centre. http://edepot.wur.nl/317612.

Bodzin, A. M. and Anastasio, D. (2006). Using Web-based GIS For Earth and Environmental Systems Education. *Journal of Geoscience Education*, 54(3):297–300. http://serc.carleton.edu/files/nagt/jge/abstracts/bodzinv54p295.pdf.

Budiu, R. (2017). Quantitative vs. Qualitative Usability Testing. https://www.nngroup.com/articles/quant-vs-qual/.

Business Analyst Learning (2018). A List of Requirements Prioritization Techniques You Should Know About. https://businessanalystlearnings.com/blog/2016/8/18/a-list-of-requirements-prioritization-techniques-you-should-know-about.

Cabaj, M. (2017). Evaluating Prototypes. Technical report. https://www.tamarackcommunity.ca/hubfs/Resources/Tools/Aid4ActionEvaluatingPrototypesMarkCabaj.pdf.

Cagle, R. F. (2003). Infrastructure Asset Management: An Emerging Direction. *AACE International Transactions*, pages PM21–PM26.

Cao, J. (2016). No Tihttps://www.uxpin.com/studio/blog/what-is-a-mockup-the-final-layer-of-ui-design/tle. https://www.uxpin.com/studio/blog/what-is-a-mockup-the-final-layer-of-ui-design/.

Catbas, F. N., Malekzadeh, M., and Khuc, T. (2013). Movable Bridge Maintenance Monitoring. Technical Report October, University of Central Florida, Orlando, Florida, U.S.A.

Cesium (2018). Cesium. https://cesiumjs.org/.

Chakravorty, D. (2018). OBJ File Format – Simply Explained for CAD and 3D Printing. https://all3dp.com/1/obj-file-format-3d-printing-cad/{#}the-obj-file-format-specification-simplified.

Chaturvedi, K. (2014). *Web based 3D analysis and visualization using HTML5 and WebGL Web based 3D analysis and visualization using*. Master's thesis, University of Twente. https://webapps.itc.utwente.nl/librarywww/papers_2014/msc/gfm/chaturvedi.pdf.

CityGML Database (2018). 3DCityDB in Action. https://www.3dcitydb.org/3dcitydb/3dcitydb-in-action.

CloudCompare (2015). Interactive Tranformation Tool. https://www.cloudcompare.org/doc/wiki/index.php?title=Interactive{_}Transformation{_}Tool.

Cozzi, P., Fili, T., and Lilley, S. (2018). Batched 3D Model. https://github.com/AnalyticalGraphicsInc/3d-tiles/tree/master/TileFormats/Batched3DModel.

de la Beaujardiere, J. (2006). OpenGIS® Web Map Server Implementation Specification. Technical report, Open Geospatial Consortium Inc. http://www.opengeospatial.org/standards/wms.

Diakite, A. (2018). About the Geo-referencing of BIM models. Technical report, Delft University of Technology, Delft, Netherlands. https://3d.bk.tudelft.nl/pdfs/18{_}georeferencing.pdf.

Dirksen, J. (2013). *Learning Three.js: The JavaScript 3D Library for WebGL*. Packt Publishing.

Dispenza, K. (2010). The Daily Life of Building Information Modeling (BIM). http://buildipedia.com/aec-pros/design-news/the-daily-life-of-building-information-modeling-bim.

Eastman, C., Teicholz, P., Sacks, R., and Liston, K. (2011). *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors*. John Wiley & Sons.

Eck, D. J. (2018). *Introduction to Computer Graphics*. Hobart and William Smith Colleges.

Erfgoed, R. v. d. C. (2014). Koninginnebrug in Rotterdam. http://rijksmonumenten.nl/monument/513925/koninginnebrug/rotterdam/.

Feuerstein, S. (2012). Bulk Processing with BULK COLLECT and FORALL. https://blogs.oracle.com/oraclemagazine/bulk-processing-with-bulk-collect-and-forall.

Fredericque, B. and Lapierre, A. (2010). The Benefits of a 3D City GIS for Sustaining City Infrastructure. *Architecture Update*, 9:64–65. https://www.scribd.com/document/50843516/benefit-of-3D-city-GIS.

Godfrey, J. (2017). *Oracle ® Application Express: Tutorial Building an Application*. Oracle. https://docs.oracle.com/database/apex-5.1/AETUT/AETUT.pdf.

Gottemoeller, F. (2014). Bridge Aesthetics: Achieving Structural Art in Bridge Design. In Chen, W.-F. and Duan, L., editors, *Bridge Engineering Handbook: Fundamentals*, pages 49–76. Taylor & Francis Group. http://civilcafe.weebly.com/uploads/2/8/9/8/28985467/bridge_engineering_handbook-_seismic_design_second_edition.pdf.

Gröger, G., Kolbe, T., Nagel, C., and Häfele, K.-H. (2012). OGC City Geography Markup Language (CityGML) Encoding Standard. *Open Geospatial Consortium*. http://www.opengis.net/spec/citygml/2.0.

Harley, A. (2018). UX Expert Reviews. https://www.nngroup.com/articles/ux-expert-reviews/.

Hida, S. A. (2014). Bridge Aesthetics: Achieving Structural Art in Bridge Design. In Chen, W.-F. and Duan, L., editors, *Bridge Engineering Handbook: Fundamentals*, pages 131–141. Taylor & Francis Group, 2nd edition. http://freeit.free.fr/Bridge%20Engineering%20HandBook/ch03.pdf.

Hixon, C. (2012). The Use of 3D-GIS Applications for Planning and Design. http://www.gis-t.org/files/EGzLi.pdf.

IGN (2015). iTowns. http://www.ign.fr/institut/innovation/itowns.

Interaction Design Foundation (2018). User Interface (UI) Design. https://www.interaction-design.org/literature/topics/ui-design.

ISO10303-242 (2014). Industrial automation systems and integration – Product data representation and exchange – Part 242: Application protocol: Managed model-based 3D engineering. https://www.iso.org/standard/57620.html.

iTowns (2017). iTowns. http://www.itowns-project.org/.

Ivory, M. Y. (2003). *Automated Web Site Evaluation: Researchers' and Practioners' Perspectives*. Springer Science+Business Media.

Jeberson Retna Raj, R. and Sasipraba, T. (2010). Disaster management system based on GIS web services. In *Proceedings of the Recent Advances in Space Technology Services and Climate Change (RSTSCC)*, Chennai, India. IEEE.

Jennings, T. (2013). *Oracle ® Application Express End User 's Guide*. Oracle. https://docs.oracle.com/database/121/AEEUG/E29541-04.pdf.

Kantor, I. (2018). Coordinates. https://javascript.info/coordinates.

Karan, E. P., Irizarry, J., and Haymaker, J. (2015). BIM and GIS Integration and Interoperability Based on Semantic Web Technology. *Journal of Computing in Civil Engineering*, 30(3):04015043. http://ascelibrary.org/doi/10.1061/{%}28ASCE{%}29CP.1943-5487.0000519.

Khronos Group Inc. (2018a). COLLADA Overview. https://www.khronos.org/collada/.

Khronos Group Inc. (2018b). glTF Overview. https://www.khronos.org/gltf/.

Koglin, T. L. (2003). *Movable Bridge Engineering*. John Wiley & Sons.

Kolár, J. and Wen, W. (2009). Visualization Aided Georeferencing of Individual 3d Models. Technical report, Aalborg University, Aalborg, Denmark. http://geovisualisierung.net/geoviz{_}hamburg/papers/09{_}4{_}Kolar.pdf.

Kolbe, T. H. (2009). Representing and Exchanging 3D City Models with CityGML. In Lee, J. and Zlatanova, S., editors, *3D Geo-Information Sciences*, pages 15–31, Seoul, Korea. Springer Verlag. http://link.springer.com/10.1007/978-3-540-87395-2{_}2.

LaViola, J. J., Kruijff, E., McMahan, R. P., Bowman, D. A., and Poupyrev, I. (2017). *3D User Interfaces: Theory and Practice*. Addison-Wesley, USA.

Lazar, J. (2001). *User-Centered Web Development*. Jones and Bartlett Publishers.

Loechel, A. and Schmid, S. (2012). Caching techniques for high - performance Web Map Services. In Gensel, J., Josselin, D., and Vandenbroucke, D., editors, *Proceedings of the AGILE'2012 International Conference on Geographic Information Science*, pages 24–27. https://agile-online.org/conference_paper/cds/agile_2012/proceedings/papers/paper_loechel_caching_techniques_for_high-performance_web_map_services_2012.pdf.

Makkonen, S. (2016). *Semantic 3D modelling for infrastructure asset management*. Master's thesis, Aalto University. https://aaltodoc.aalto.fi/handle/123456789/23659.

Mao, B. (2011). *Visualisation and Generalisation of 3D City Models*. Phd thesis, Royal Institute of Technology (KTH). http://www.diva-portal.org/smash/get/diva2:456906/FULLTEXT02.pdf.

Marel, H. V. D. (2014). Reference Systems for Surveying and Mapping. Technical report, Delft University of Technology, Delft, Netherlands. http://gnss1.tudelft.nl/pub/vdmarel/reader/CTB3310{_}RefSystems{_}1-2a{_}online.pdf.

Masirin, H. M. I. B. M. and Zain, R. B. M. (2013). Overview and Preliminary Study of Approach - Slab Design Concept for Bridges. *Procedia Engineering*, 54:774–784. http://dx.doi.org/10.1016/j.proeng.2013.03.071.

Matsuda, K. and Lea, R. (2013). *WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL*. Pearson Education Inc.

Mendez, E., Schall, G., Havemann, S., Fellner, D., Schmalstieg, D., and Junghanns, S. (2008). Generating Semantic 3D Models of Underground Infrastructure. *IEEE Computer Graphics and Applications*, 28(3):48–57. https://www.researchgate.net/publication/3210666_Generating_Semantic_3D_Models_of_Underground_Infrastructure.

Miettinen, R. and Paavola, S. (2014). Beyond the BIM utopia: Approaches to the development and implementation of building information modeling. *Automation in Construction*, 43:84–91. http://dx.doi.org/10.1016/j.autcon.2014.03.009.

Mozilla (2018a). click | Properties. https://developer.mozilla.org/en-US/docs/Web/Events/click.

Mozilla (2018b). HTTP response status codes. https://developer.mozilla.org/de/docs/Web/HTTP/Status.

Mozilla (2018c). Sending form data. https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Sending{_}and{_}retrieving{_}form{_}data.

NAMS and IPWEA (2011). International Infrastructure Management Manual (IIMM). Technical report, New Zealand National Asset Management Steering Group and Institute of Public Works Engineering Australia.

NEN (2018a). NEN 2767-4-2: Beheerobjecten. https://www.nen2767-4.nl/beheerobject.html.

NEN (2018b). NEN 2767-4 Conditiemeting van Infrastructuur. https://www.nen2767-4.nl/.

Neuman, W. L. (2014). *Social Research Methods: Qualitative and Quantitative Approaches*. Pearson Education. http://letrunghieutvu.yolasite.com/resources/w-lawrence-neuman-social-research-methods{_}-qualitative-and-quantitative-approaches-pearson-education-limited-2013.pdf.

Open Source Geospatial Foundation (2017). Services. http://docs.geoserver.org/2.12.2/user/services/index.html{#}services.

Oracle (2005). The htp and htf Packages. https://docs.oracle.com/cd/B14099{_}19/web.1012/b15896/pshtp.htm.

Oracle (2015). Understanding Application Processes. https://docs.oracle.com/html/E39147{_}04/bldr{_}app{_}proc.htm.

Oracle (2018). Why Oracle APEX. https://apex.oracle.com/en/platform/why-oracle-apex/.

OSM Buildings (2018). OSM Buildings Solutions. https://osmbuildings.org/solutions/.

Paiva, A. C. and Baptista, C. S. (2009). Web-Based GIS. In Khosrow-Pour, M., editor, *Encyclopedia of Information Science and Technology*, pages 4053–4057. Information Science Reference.

Petrus, T. (2013). jQuery with APEX. In Scott, J., Buytaert, N., Cannell, K., D'Souza, M., Gault, D., Gielis, D., Hartman, R., Kubicek, D., Mattamal, R., McGhan, D., Mignault, F., Petrus, T., Rimblas, J., and Ruepprich, C., editors, *Expert Oracle Application Express*, pages 455–476. Springer Science+Business Media. http://medcontent.metapress.com/index/A65RM03P4874243N.pdf{%}5Cnhttp://books.google.com/books?hl=en{&}lr={&}id=bMmb{_}GddSnEC{&}oi=fnd{&}pg=PP3{&}dq=Expert+Oracle+Application+Express+Security{&}ots=meu{_}wGYtYm{&}sig=IAFMq1rfdbh9gbgdL6FW1lKCcDA{%}5Cnhttp://books.google.com/boo.

Poppe, M. (2017). Three.js-City. https://github.com/mauriciopoppe/Three.js-City.

Prandi, F., Devigili, F., Soave, M., Di Staso, U., and De Amicis, R. (2015). 3D web visualization of huge cityGML models. In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, volume 40, pages 601–605, La Grande Motte, France. Copernicus GmbH, Göttingen.

Royal Observatory of Belgium (2012). Coordinate Systems. http://gnss.be/systems{_}tutorial.ph.

Rubin, J. and Chisnell, D. (2008). *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*. Wiley Publishing, 2nd edition.

Sanz Subirana, J., Juan Zornoza, J. M., and Hernández, M. (2011a). Ellipsoidal and Cartesian Coordinates Conversion. https://gssc.esa.int/navipedia/index.php/Ellipsoidal{_}and{_}Cartesian{_}Coordinates{_}Conversion.

Sanz Subirana, J., Juan Zornoza, J. M., and Hernández, M. (2011b). Satellite Coordinates. https://gssc.esa.int/navipedia/index.php/Satellite{_}Coordinates.

Sayar, A. and Pierce, M. (2005). OGC Compatible Geographical Information Systems Web Services. Technical report, Indiana University. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.62.2581{&}rep=rep1{&}type=pdf.

Schilling, A., Bolling, J., and Nagel, C. (2016). Using glTF for streaming CityGML 3D city models. In *Proceedings of the 21st International Conference on Web3D Technology - Web3D '16*, number October 2015, pages 109–116, Anaheim, CA, USA. ACM New York, NY, USA. http://dl.acm.org/citation.cfm?doid=2945292.2945312.

Scianna, A. (2013). Building 3D GIS data models using open source software. *Applied Geomatics*, 5(2):119–132. http://link.springer.com/10.1007/s12518-013-0099-3.

Siniscalco, M. T. and Auriat, N. (2005). Questionnaire design. In Ross, K. N., editor, *Quantitative research methods in educational planning*, pages 22–35. UNESCO International Institute for Educational Planning. http://unesdoc.unesco.org/images/0021/002145/214555E.pdf.

Skinner, G. D. and Rowe, R. K. (2004). Design and behaviour of a geosynthetic reinforced retaining wall and bridge abutment on a yielding foundation. *Geotextiles and Geomembranes*, 23(3):234–260. https://www.researchgate.net/publication/229372158_Design_and_behavior_of_a_geosynthetic_reinforced_retaining_wall_and_bridge_abutment_on_a_yielding_foundation.

Soegaard, M. and Dam, R. F. (2013). *The Encyclopedia of Human-Computer Interaction*. The Interaction Design Foundation.

Sommerville, I. (2010). *Software Engineering*. Addison-Wesley, USA, 9th edition. https://edisciplinas.usp.br/pluginfile.php/2150022/mod_resource/content/1/1429431793.203Software%20Engineering%20by%20Somerville.pdf.

Stadler, A. and Kolbe, T. H. (2007). Spatio-Semantic Coherence in the Integration of 3D City Models. In *Proceedings of the 5th International ISPRS Symposium on Spatial Data Quality*, pages 13–15. ISPRS. http://www.isprs.org/proceedings/XXXVI/2-C43/Session1/paper{_}Stadler.pdf.

Succar, B., Sher, W., and Williams, A. (2013). An integrated approach to BIM competency assessment, acquisition and application. *Automation in Construction*, 35:174–189. http://dx.doi.org/10.1016/j.autcon.2013.05.016.

Taalman, L. (2015). three.js Wisdom Collector. http://mathgrrl.com/hacktastic/wisdom-collectors/three-js-collector/.

Tatarević, V. (2007). Understanding and Creating Web GIS. *Ekscentar*, 9. https://hrcak.srce.hr/index.php?show=clanak&id_clanak_jezik=19243.

Tekla (2018). Products. https://www.tekla.com/products.

Tepas, A. (2014). Open IT-architectuur zet beheer publieke & private buitenruimte weer op de kaart. https://www.slideshare.net/AMTepas/open-architectuur.

Three.js (2018). Three.js Documentation. https://threejs.org/docs/.

Tidwell, J. (2010). *Designing Interfaces: Patterns for Effective Interaction Design*. O'Reilly Media.

Too, E. G. (2008). A Framework for Strategic Infrastructure Asset Management. In *Third World Congress on Engineering Asset Management and Intelligent Maintenance Systems Conference*, pages 1–10, Beijing, China. Springer. https://www.researchgate.net/publication/27475965_A_Framework_for_Strategic_Infrastructure_Asset_Management.

Tu, S., Flanagin, M., Wu, Y., Abdelguerfi, M., Normand, E., and Mahadevan, V. M. V. (2004). Design strategies to improve performance of GIS Web services. *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.*, 2:1–5. https://www.researchgate.net/publication/4068206_Design_strategies_to_improve_performance_of_GIS_Web_services.

van Loenen, B. (2009). Developing geographic information infrastructures: The role of information policies. *International Journal of Geographical Information Science*, 23(2):195–212. https://www.researchgate.net/publication/34668150_Developing_geographic_information_infrastructures_the_role_of_information_policies.

van Oosterom, P., Quak, W., Tijssen, T., and Verbree, E. (2000). The Architecture of the Geo-Information Infrastructure. In *Proceedings of the UDMS 2000, 22nd Urban Data Management Symposium*, pages 9–18, Delft, Netherlands. Urban Data Management Society. http://sdistandards.icaci.org/wp-content/uploads/2014/10/MAfA{_}SectionC{_}Integrated{_}V10.pdf.

Vanlande, R., Nicolle, C., and Cruz, C. (2008). IFC and Buildings Lifecycle Management. *Automation in Construction*, 18(1):70–78. https://www.sciencedirect.com/science/article/pii/S0926580508000800.

Voldan, P. (2012). Developing web map application based on user centered design. *Geoinformatics FCE CTU*, 7:131–141. https://ojs.cvut.cz/ojs/index.php/gi/article/view/2617.

Vretanos, P. A. (2005). OpenGIS ® Web Feature Service Implementation Specification. Technical report, Open Geospatial Consortium Inc. https://www.opengeospatial.org/standards/wfs.

W3Schools (2018). XML HttpRequest. https://www.w3schools.com/xml/xml{_}http.asp.

Wang, Y., Huynh, G., and Williamson, C. (2013). Integration of Google Maps/Earth with microscale meteorology models and data visualization. *Computers and Geosciences*, 61(2013):23–31. http://dx.doi.org/10.1016/j.cageo.2013.07.016.

Whiteside, A. and Evans, J. D. (2007). OpenGIS® Web Coverage Service Implementation Specification. Technical report, Open Geospatial Consortium Inc . https://www.opengeospatial.org/standards/wcs.

Wolter, S. (2010). Life Cycle Asset Management. *Life Cycle Engineering*, pages 1–7. https://www.lce.com/pdfs/LCAM-Whitepaper-204.pdf.

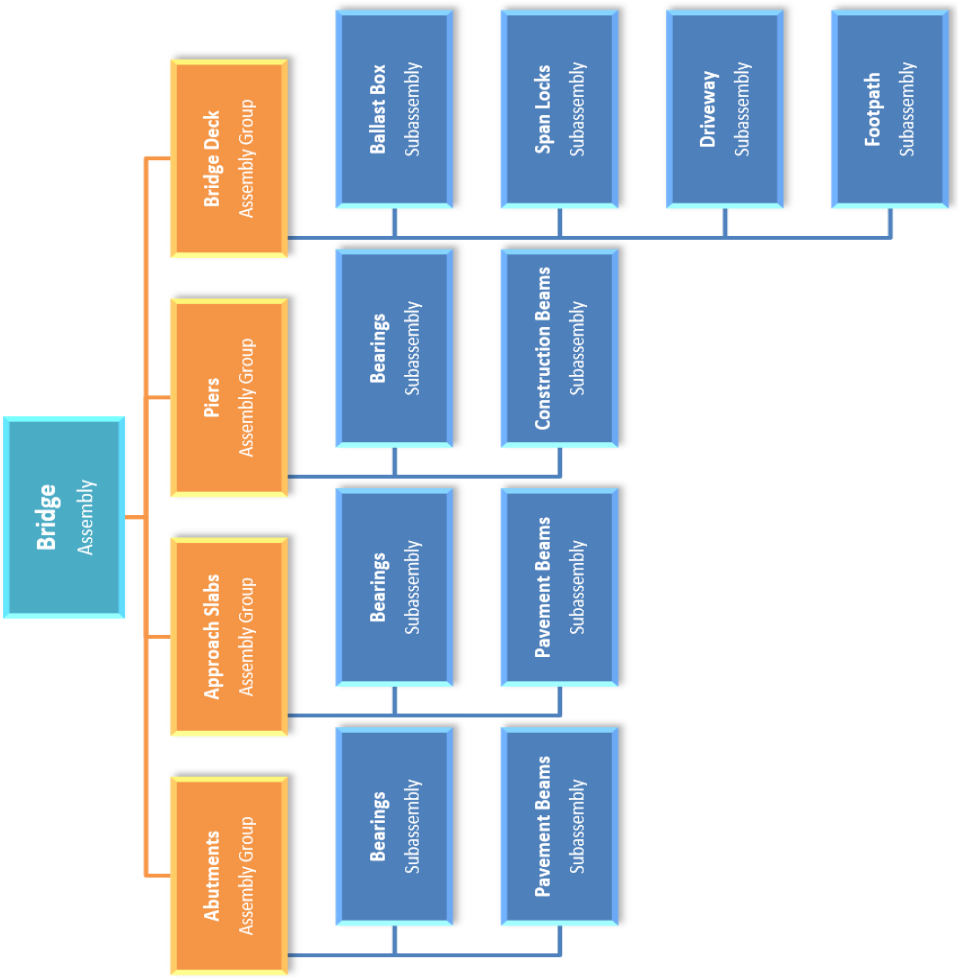
Yang, C., Wu, H., Huang, Q., Li, Z., Li, J., Li, W., Miao, L., and Sun, M. (2011). WebGIS performance issues and solutions. In Li, S., Dragicevic, S., and Veenendaal, B., editors, *Advances in Web-based GIS, Mapping Services and Applications*, pages 121–138. Taylor & Francis Group. https://www.researchgate.net/publication/267388803_WebGIS_performance_issues_and_solutions.

Yoders, J. (2013). BIM Details: How Levels of Development Provide Model Clarity. https://www.autodesk.com/redshift/levels-of-development-provide-3d-model-clarity/.

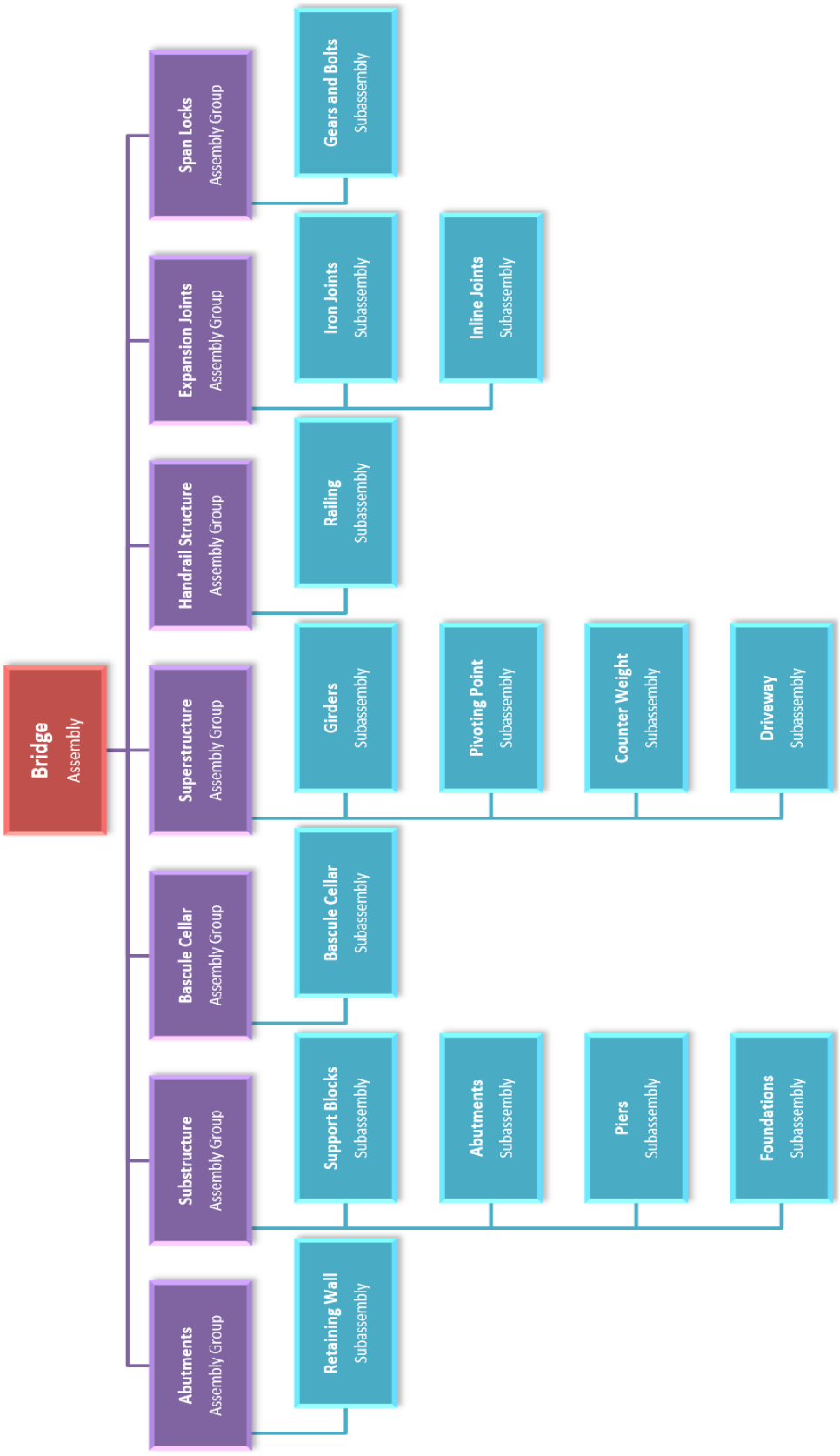
Zlatanova, S., Rahman, A., and Pilouk, M. (2002). Trends in 3D GIS development. *Journal of Geospatial*, 4(2):71–80. https://www.researchgate.net/publication/228763906_Trends_in_3D_GIS_development.

Zlatanova, S., Stoter, J., and Isikdag, U. (2012). Standards for Exchange and Storage of 3D Information: Challenges and Opportunities for Emergency Response. In *Proceedings of the 4th International Conference on Cartography & GIS*, pages 17 – 28. International Cartographic Association. https://repository.tudelft.nl/islandora/object/uuid:a7d1e4a4-3155-43ce-81d4-6ad6cfc26617?collection=research.

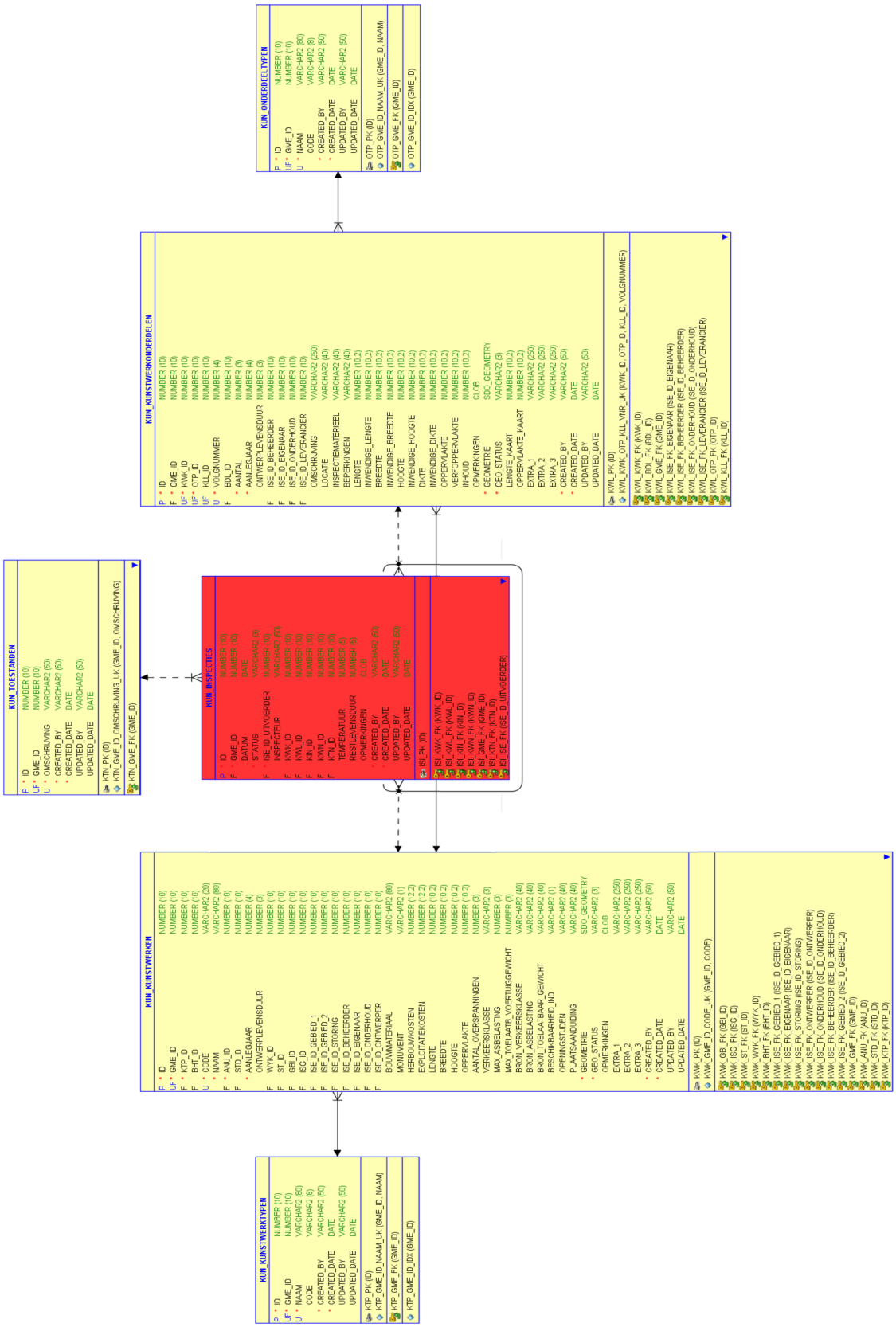
A BIM MODEL COMPOSITION



B | BIM MODEL DECOMPOSITION



C UML DIAGRAM OF THE DATABASE STRUCTURE



D HTML, CSS AND JAVASCRIPT CODES

Listing D.1: HTML code of the 3D viewer configuration

```
<!DOCTYPE html>
<html>
   <head>
     <meta charset="UTF-8"/>
     <link href="3dviewer.css">
     <link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
     <link href="https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/css/font-awesome.min.css">
     <link href="https://fonts.googleapis.com/icon?family=Material+Icons">
     <link href="https://code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css">
     <link href="https://code.jquery.com/ui/1.12.1/themes/smoothness/jquery-ui.css">
     <link href="https://code.jquery.com/ui/1.10.4/themes/flick/jquery-ui.css">
   </head>
   <body>
     <div id="grid">
       <div class="openLegend"><i class="material-icons">dvr</i></div>
       <div class="closeLegend"><i class="material-icons">dvr</i></div>
       <div class="openSwitcher"><i class="material-icons">layers</i></div>
       <div class="closeSwitcher"><i class="material-icons">layers</i></div>
       <div class="openHistory"><i class="material-icons" id="icon1">query_builder</i></div>
       <div class="closeHistory"><i class="material-icons" id="icon1">query_builder</i></div>
       <div class="tabLegend">
         <div class="panel-heading"><h4 class="panel-title"><span class="material-icons">dvr</span>
              Legenda</h4></div>
           <div id="tab1">
             <div class="panel-body">
               <table>
                 <tr><td class="inputrows" id="i1"><div id="first"></div></td><td><input type="text"
                      class="status" id="color1" value="1 - Uitstekende conditie"></td></tr>
                 <tr><td class="inputrows"><div id="second"></div></td><td><input type="text" class="
                      status" id="color2" value="2 - Goede conditie"></td></tr>
                 <tr><td class="inputrows"><div id="third"></div></td><td><input type="text" class="
                      status" id="color3" value="3 - Redelijke conditie"></td></tr>
                 <tr><td class="inputrows"><div id="fourth"></div></td><td><input type="text" class="
                      status" id="color4" value="4 - Matige conditie"></td></tr>
                 <tr><td class="inputrows"><div id="fifth"></div></td><td><input type="text" class="
                      status" id="color5" value="5 - Slechte conditie"></td></tr>
                 <tr><td class="inputrows" id="i6"><div id="sixth"></div></td><td><input type="text"
                      class="status" id="color6" value="6 - Zeer slechte conditie"></td></tr>
               </table>
             </div>
           </div>
         </div>
       </div>
       <div class="tabSwitcher">
         <div class="panel-heading"><h4 class="panel-title"><span class="material-icons">layers</span>
              Switcher</h4></div>
           <div id="tab2">
             <div class="panel-body">
               <form>
                 <fieldset>
```

```
                <div class="content">
                  <div>
                    <label class="elements">Kerende constructie
                      <input type="checkbox" class="models" onclick="showObject()" checked>
                      <span class="checkmark"></span>
                    </label>
                  </div>
                  <div>
                    <label class="elements">Steunpunt
                      <input type="checkbox" class="models" onclick="showObject()" checked>
                      <span class="checkmark"></span>
                    </label>
                  </div>
                  <div>
                    <label class="elements">Hoofddraagconstructie
                      <input type="checkbox" class="models" onclick="showObject()" checked>
                      <span class="checkmark"></span>
                    </label>
                  </div>
                  <div>
                    <label class="elements">Aandrijving en bewegingswerk
                      <input type="checkbox" class="models" onclick="showObject()" checked>
                      <span class="checkmark"></span>
                    </label>
                  </div>
                </div>
              </fieldset>
            </form>
          </div>
        </div>
      </div>
      <div class="tabHistory">
        <div class="panel-heading"><h4 class="panel-title"><span class="material-icons" id="icon1">
            query_builder</span>Geschiedenis</h4></div>
          <div id="tab3">
            <div class="panel-body">
              <form>
                <fieldset>
                  <div class="content">
                    <div>
                      <label class="years">2012
                        <input type="radio" name="condition" id="cng1" onclick="showHistory()" checked>
                        <span class="checkmark"></span>
                      </label>
                    </div>
                    <div>
                      <label class="years">2007
                        <input type="radio" name="condition" id="cng2" onclick="showHistory()">
                        <span class="checkmark"></span>
                      </label>
                    </div>
                    <div>
                      <label class="years">2002
                        <input type="radio" name="condition" id="cng3" onclick="showHistory()">
                        <span class="checkmark"></span>
                      </label>
                    </div>
                  </div>
                </fieldset>
              </form>
            </div>
```

```
104             </div>
105           </div>
106         </div>
107         <div>
108           <label class="toggleSwitch">
109             <input type="checkbox" onclick="showGround()" checked>
110             <span class="slider round"></span>
111           </label>
112         </div>
113         <div id="info">
114           <table id="table1">
115             <p class="header"><span class="fa fa-info-circle" id="icon3"></span>Details<br style="clear:
                    both;"/></p>
116             <tr><td class="inputrows" id="first">Beheerobject</td><td><input type="text" id="beheerobj">
                    </td></tr>
117             <tr><td class="inputrows">Vakdiscipline</td><td><input type="text" id="discipline"></td></tr>
118             <tr><td class="inputrows">Element</td><td><input type="text" id="element"></td></tr>
119             <tr><td class="inputrows">Inspectiedatum</td><td><input type="text" id="inspdate"></td></tr>
120             <tr><td class="inputrows">Inspecteur</td><td><input type="text" id="inspector"></td></tr>
121             <tr><td class="inputrows">Opmerkingen</td><td><input type="text" id="notes"></td></tr>
122           </table>
123           <table id="table2">
124             <tr><td id="year3"><p class="scores" id="score3"></p></td><td id="year2"><p class="scores"
                      id="score2"></p> </td><td id="year1"><p class="scores" id="score1"></p></td></tr>
125           </table>
126           <div id="line"></div>
127           <table id="table3">
128             <tr><td id="label3">2002</td><td id="label2">2007</td><td id="label1">2012</td></tr>
129           </table>
130         </div>
131         <div class="progress">
132           <div class="progress-bar progress-bar-striped progress-bar-animated" style="width:0%"></div>
133         </div>
134         <div id="webgl"><div>
135       </div>
136       <script src="3dviewer.js"></script>
137       <script src="three.js"></script>
138       <script src="OrbitControls.js"></script>
139       <script src="MTLLoader.js"></script>
140       <script src="OBJLoader.js"></script>
141       <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
142       <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>
143       <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
144     </body>
145 </html>
```

Listing D.2: CSS styling of the 3D viewer configuration

```
1  body {
2     margin: 0 auto;
3     padding: 0 auto;
4     overflow: hidden;
5  }
6
7  #grid {
8     position: relative;
9     width: auto;
10    height: 592px;
11    margin: 0 auto;
12    padding: 0 auto;
13 }
```

```
14
15 .openLegend,
16 .closeLegend,
17 .openSwitcher,
18 .closeSwitcher,
19 .openHistory,
20 .closeHistory {
21    display: block;
22    position: absolute;
23    z-index: 3;
24    width: 36px;
25    height: 36px;
26    top: 10px;
27    background-color: #F5F5F5;
28    background-clip: padding-box;
29    color: #2196F3;
30    border: 1px solid #DDD;
31    border-radius: 4px;
32    -moz-border-radius: 4px;
33    opacity: .9;
34    filter: alpha(opacity=.9);
35    cursor: pointer;
36 }
37
38 .openLegend,
39 .closeLegend {
40    left: 12px;
41 }
42
43 .openSwitcher,
44 .closeSwitcher {
45    left: 56px;
46 }
47
48 .openHistory,
49 .closeHistory {
50    left: 102px;
51 }
52
53 .closeLegend,
54 .closeSwitcher,
55 .closeHistory {
56    display: none;
57 }
58
59 .openLegend > i,
60 .closeLegend > i,
61 .openSwitcher > i,
62 .closeSwitcher > i,
63 .openHistory > i,
64 .closeHistory > i,
65 .openLegend > i:focus,
66 .closeLegend > i:focus,
67 .openSwitcher > i:focus,
68 .closeSwitcher > i:focus,
69 .closeHistory > i:focus,
70 .openHistory > i:focus,
71 .openLegend > i:hover,
72 .closeLegend > i:hover,
73 .openSwitcher > i:hover,
74 .closeSwitcher > i:hover,
75 .openHistory > i:hover,
```

```
.closeHistory > i:hover {
   margin-top: 8px;
   margin-left: 8px;
   font-size: 1.9rem;
   color: #2196F3;
}

.panel-body {
   padding: 0 !important;
   margin: 0 !important;
}

.panel-heading {
   height: 36px;
   margin: 0 !important;
   padding: 0 !important;
   font-weight: 610px;
}

.panel-heading .material-icons {
   margin-right: 8px;
   vertical-align: middle;
   line-height: 1.75;
   text-transform: none;
}

.panel-heading h4 {
   margin-top: 0.1em;
   margin-left: 0.002em;
}

.panel-title {
   height: 36px;
   margin: 0 !important;
   padding-left: 10px;
   background-color: #F5F5F5;
   color: #2196F3;
   border-color: #DDDDDD;
   border-width: 1px;
   line-height: 1.5;
}

.material-icons {
   margin-top: -3px;
}

.tabLegend,
.tabSwitcher,
.tabHistory {
   display: none;
   position: absolute;
   z-index: 3;
   width: 215px;
   top: 10px;
   left: 10px;
   margin: 0 !important;
   padding: 0 !important;
   background-color: white;
   background-clip: padding-box;
   border: 1px solid #DDD;
   border-radius: 4px;
   -moz-border-radius: 4px;
```

```
    box-shadow: 0 6px 12px rgba(0,0,0,.175);
    -webkit-box-shadow: 0 6px 12px rgba(0,0,0,.175);
    -moz-box-shadow: 0 6px 12px rgba(0,0,0,.175);
    opacity: .9;
    filter: alpha(opacity=.9);
}

#tab1 table {
    width: 170px;
    margin: 3.5px 0px 4px 0px;
}

#tab1 td {
    padding-top: 3px;
    padding-bottom: 3px;
}

#tab1 .inputrows {
    text-align: left;
    width: 10%;
    padding-right: 8px;
    padding-left: 12px;
}

#tab1 #first, #second, #third, #fourth, #fifth, #sixth {
    float: left;
    margin: 0 auto;
    padding-top: 0px;
    padding-right: 5px;
    width: 18px;
    height: 18px;
}

#tab1 #first {
    background: #2FC02F;
}

#tab1 #second {
    background: #9ACD32;
}

#tab1 #third {
    background: #FFDF00;
}

#tab1 #fourth {
    background: #FF8C00;
}

#tab1 #fifth {
    background: #FF0000;
}

#tab1 #sixth {
    background: #CC0000;
}

#tab1 #color1, #color2, #color3, #color4, #color5, #color6 {
    width: 160px;
    height: 25px;
    font-size: 1.2rem;
    border: none;
```

```
}

#tab2 label,
#tab3 label {
   display: block;
   margin-top: 10px;
   margin-bottom: 10px;
   font-size: 1.2rem;
}

.content {
   padding-right: 5px;
   padding-left: 12px;
}

.elements,
.years {
   position: relative;
   margin-bottom: 12px;
   padding-left: 30px;
   font-weight: normal;
   -webkit-user-select: none;
   -moz-user-select: none;
   -ms-user-select: none;
   user-select: none;
   cursor: pointer;
}

.elements input,
.years input {
   position: absolute;
   opacity: 0;
   cursor: pointer;
}

.elements .checkmark,
.years .checkmark {
   position: absolute;
   width: 18px;
   height: 18px;
   top: 0;
   left: 0;
   background-color: #eee;
}

.elements input:checked ~ .checkmark,
.years input:checked ~ .checkmark {
   background-color: #2196F3;
}

.elements .checkmark:after,
.years .checkmark:after {
   content: "";
   position: absolute;
   display: none;
}

.elements input:checked ~ .checkmark:after,
.years input:checked ~ .checkmark:after {
   display: block;
}

```

```
.elements .checkmark:after,
.years .checkmark:after {
   width: 5.5px;
   height: 9px;
   left: 6.4px;
   top: 3.5px;
   border: solid white;
   border-width: 0 3px 3px 0;
   -webkit-transform: rotate(45deg);
   -ms-transform: rotate(45deg);
   transform: rotate(45deg);
}

.toggleSwitch {
   display: inline-block;
   position: absolute;
   width: 50px;
   height: 30px;
   z-index: 4;
   top: 11px;
   left: 48.5%;
   right: 48.5%;
}

.toggleSwitch input {
   display: none;
   width: 0;
   height: 0;
   opacity: 0;
}

.slider {
   position: absolute;
   top: 0;
   left: 0;
   right: 0;
   bottom: 0;
   background-color: #ccc;
   -webkit-transition: .4s;
   transition: .4s;
   cursor: pointer;
}

.slider:before {
   content: "";
   position: absolute;
   width: 21.6px;
   height: 21.6px;
   left: 3.4px;
   bottom: 4px;
   background-color: white;
   -webkit-transition: .4s;
   transition: .4s;
}

input:checked + .slider {
   background-color: #2196F3;
}

input:focus + .slider {
   box-shadow: 0 0 1px #2196F3;
}
```

```
324
325 input:checked + .slider:before {
326    -webkit-transform: translateX(21.6px);
327    -ms-transform: translateX(21.6px);
328    transform: translateX(21.6px);
329 }
330
331 .slider.round {
332    border-radius: 34px;
333 }
334
335 .slider.round:before {
336    border-radius: 50%;
337 }
338
339 #info {
340    display: none;
341    position: absolute;
342    z-index: 3;
343    width: auto;
344    height: 335px;
345    top: 12px;
346    right: 12px;
347    margin: 0 !important;
348    padding: 0 !important;
349    background-color: white;
350    background-clip: padding-box;
351    border: 1px solid #ddd;
352    border-radius: 4px;
353    -moz-border-radius: 4px;
354    box-shadow: 0 6px 12px rgba(0,0,0,.175);
355    -webkit-box-shadow: 0 6px 12px rgba(0,0,0,.175);
356    -moz-box-shadow: 0 6px 12px rgba(0,0,0,.175);
357    opacity: .9;
358    filter: alpha(opacity=.9);
359 }
360
361 #info table {
362    width: 240px;
363    margin: 0px 5px 0px 0px;
364    padding: 0 !important;
365 }
366
367 #info .inputrows {
368    width: 10%;
369    margin-bottom: 5px;
370    padding-right: 9px;
371    padding-left: 8.5px;
372    text-align: right;
373    font-size: 1.2rem;
374 }
375
376 #info #first {
377    padding-top: 5px;
378 }
379
380 #info #discipline, #beheerobj, #element, #inspdate, #inspector, #notes {
381    width: 136px;
382    height: auto;
383    margin-top: 5px;
384    margin-bottom: 5px;
385    padding: 2px 5px 2px 5px;
```

```
    font-size: 1.16rem;
}

#info #beheerobj {
    margin-top: 9px;
}

#info #table2 {
    width: 210px;
    margin-left: 25px;
}

#info #table2 td {
    width: 15px;
    height: 15px;
    margin: 0 !important;
    padding-top: 5px;
}

#info #table2 #year3 {
    padding-left: 20px;
    padding-right: 2px;
}

#info #table2 #year2 {
    padding-right: 9.5px;
}

#info #table2 #year1 {
    padding-right: 15px;
}

#info #table3 {
    width: 210px;
    margin-left: 30px;
}

#info #table3 td {
    margin: 0 !important;
    padding-top: 1px;
    font-size: 1.1rem;
}

#info #table3 #label3 {
    padding-left: 14.5px;
    padding-right: 2.8px;
}

#info #table3 #label2 {
    padding-right: 7.9px;
}

#info #table3 #label1 {
    padding-right: 15px;
}

#info .scores {
    position: relative;
    width: 12px;
    height: 12px;
    margin: 0 !important;
    padding: 12px;
```

```
    background: #f3961c;
    background: linear-gradient(top, #f9d835, #f3961c);
    border: 2px solid grey;
    border-radius: 3.5px;
}

#info .scores:after {
    content: "";
    display: block;
    position: absolute;
    width: 0;
    left: 6.2px;
    bottom: -8px;
    border-width: 6.9px 6.9px 0;
    border-style: solid;
    border-color: grey transparent;
}


#line {
    width: 190px;
    margin-top: 6px;
    margin-left: 24px;
    border-bottom: 1px solid grey;
}

.progress {
    display: block;
    position: absolute;
    z-index: 3;
    width: 250px;
    height: 12px;
    left: 6.5px;
    bottom: 7px;
    background-color: #F5F5F5;
    background-clip: padding-box;
    border: 1px solid #DDD;
    border-radius: 5px;
    -moz-border-radius: 5px;
    box-shadow: 0 6px 12px rgba(0,0,0,.175);
    -webkit-box-shadow: 0 6px 12px rgba(0,0,0,.175);
    -moz-box-shadow: 0 6px 12px rgba(0,0,0,.175);
    opacity: .9;
    filter: alpha(opacity=.9);
}

.progress-bar {
    z-index: 3;
    width: 250px;
    height: 12px;
    left: 6.5px;
    bottom: 7px;
    background-color: #2196F3;
    background-clip: padding-box;
    border: none;
    border-radius: 5px;
    -moz-border-radius: 5px;
    box-shadow: 0 6px 12px rgba(0,0,0,.175);
    -webkit-box-shadow: 0 6px 12px rgba(0,0,0,.175);
    box-shadow: 0 6px 12px rgba(0,0,0,.175);
    -moz-box-shadow: 0 6px 12px rgba(0,0,0,.175);
    opacity: .9;
```

```
   filter: alpha(opacity=.9);
}

#webgl {
   display: block;
   position: absolute;
   z-index: 1;
   width: 100% !important;
   height: 100% !important;
   margin: 0 auto;
   padding: 0 auto;
   top: 0;
   left: 0;
}

:-moz-full-screen canvas {
   position: absolute;
   width: 100%;
   height: 100%;
   top: 0;
   left: 0;
   right: 0;
   bottom: 0;
   margin: auto;
   background-color: white;
   box-shadow: 0px 0px 4px white;
}

#3d_viewer > .t-Region-bodyWrap > .t-Region-body {
   overflow: hidden;
   padding: 0;
}
```

Listing D.3: JavaScript code of the 3D viewer configuration

```
var dataId, objId;
var id, code, files, xhr;
var webglCanvas, canvasBounds, renderer, scene;
var camera, light, controls;
var manager, loader, percentComplete, info;
var vector, raycaster, intersects, intersected1, intersected2, intersected3;
var currentMaterial1, currentMaterial2, currentMaterial3;
var valuesBBox, groundTexture, groundMaterial, groundLength, groundGeometry, ground;
var wmsRequest;

var objects = [];

var index = 0;

var mouse = { x: 0, y: 0 };

$(document).on( "featureselected", function ( event, feature ) {

 dataId = feature.get( "ID" );

});

$(document).on( "click", ".a-IRR-table tr:not( :first-of-type )", function () {

 var isWijzigingenOk = true;

```

```
...

if ( isWijzigingenOk ) {

 $( this ).parent().find( "tr.is-selected" ).removeClass( "is-selected" );
 $( this ).addClass( "is-selected" );

 id = $( this ).find( ".details" ).attr( "data-id" );
 code = $( this ).find( ".details" ).attr( "data-code" );

 files = [ "/obsurv/kunstwerken/modelen/" + String(id) + "_Kerende constructie.obj",
           "/obsurv/kunstwerken/modelen/" + String(id) + "_Steunpunt.obj",
           "/obsurv/kunstwerken/modelen/" + String(id) + "_Hoofddraagconstructie.obj",
           "/obsurv/kunstwerken/modelen/" + String(id) + "_Aandrijving en bewegingswerk.obj" ];

 files.forEach( function( file ) {

  xhr  = new XMLHttpRequest();
  xhr.fileName = file;

  xhr.onreadystatechange = function() {

   if ( xhr.status == 404 ) {

    $( "#grid" ).hide();

   } else {

    $( "#grid" ).show();
    $( "#webgl" ).trigger( "onload" );

   }

   }

  xhr.open( "GET", file, true );
  xhr.send(null);

 });

 init();
 animate();
 render();

 manager = new THREE.LoadingManager();

 manager.onProgress = function ( file, loaded, total ) {

  percentComplete = loaded / files.length * 100;

  $( ".progress-bar" ).width( percentComplete + "%" );

 };

 manager.onError = function ( file ) {

  console.log( "Error loading " + file );

 };

 function loadFile ( scene, info ) {

```

```
loader = new THREE.OBJLoader( manager );
loader.load( files[index], function( object ) {

 object.userData.Name = files[index].split( "_" )[1].split( "." )[0];

 /*Georeferencing approach that works if reference points can be identified in the 3D model
 coordinatesWGS = [ { lat: 51.912821, lon: 4.497943 },
                    { lat: 51.913174, lon: 4.497060 },
                    { lat: 51.912480, lon: 4.498700 } ];

 coordinatesECEF0 = convertToECEF( coordinatesWGS[0].lat, coordinatesWGS[0].lon, 0 );
 coordinatesECEF1 = convertToECEF( coordinatesWGS[1].lat, coordinatesWGS[1].lon, 0 );
 coordinatesECEF2 = convertToECEF( coordinatesWGS[2].lat, coordinatesWGS[2].lon, 0 );

 var phi = THREE.Math.degToRad( coordinatesWGS[0].lat );
 var lambda = THREE.Math.degToRad( coordinatesWGS[0].lon );

 var shift = { x: coordinatesECEF1.x - coordinatesECEF0.x,
               y: coordinatesECEF1.y - coordinatesECEF0.y,
               z: coordinatesECEF1.z - coordinatesECEF0.z }

 var rotationMatrix = new THREE.Matrix4();

 rotationMatrix.set(
   - Math.sin(lambda), Math.cos(lambda), 0, 0,
   - Math.sin(phi) * Math.cos(lambda), - Math.sin(phi) * Math.sin(lambda), Math.cos(phi), 0,
   Math.cos(phi) * Math.cos(lambda), Math.cos(phi) * Math.sin(lambda), Math.sin(phi), 0,
   0, 0, 0, 1
 );

 var translationMatrix = new THREE.Matrix4();
 translationMatrix.set(
   1, 0, 0, shift.x,
   0, 1, 0, shift.y,
   0, 0, 1, shift.z,
   0, 0, 0, 1
 );

 object.applyMatrix( rotationMatrix.multiply( translationMatrix ) );*/

 var transformationMatrix = new THREE.Matrix4();
 transformationMatrix.set(
   0.016862, 0.018996, 0, 93811.90625,
   -0.018996, 0.016862, 0, 436408.6875,
   0, 0, 0.0254, 0,
   0, 0, 0, 1
 );
 object.applyMatrix( transformationMatrix );

 if ( index == 0 ) {

  valuesBBox = makeBBox( object );
  createGround( valuesBBox, object.rotation );

 }

 object.up.set( 0, 0, 1 );

 object.position.x = 0;
 object.position.y = 0;

 object.updateMatrixWorld();
```

```
151
152     scene.add( object );
153
154     objects.push( object );
155
156     for ( var i = 0; i < objects.length; i++ ) {
157
158      objects[i].userData.DatabaseId = info[i].ElementID;
159
160      objects[i].userData.InspectionDate12 = info[i].InspDate;
161      objects[i].userData.Inspector12 = info[i].Inspector;
162      objects[i].userData.Score12 = info[i].Score;
163
164      objects[i].userData.InspectionDate07 = info[i+4].InspDate;
165      objects[i].userData.Inspector07 = info[i+4].Inspector;
166      objects[i].userData.Score07 = info[i+4].Score;
167
168      objects[i].userData.InspectionDate02 = info[i+8].InspDate;
169      objects[i].userData.Inspector02 = info[i+8].Inspector;
170      objects[i].userData.Score02 = info[i+8].Score;
171
172      objects[i].traverse( function ( child ) {
173
174       if ( child instanceof THREE.Mesh ) {
175
176        colorComponent( child, objects[i].userData.Score12 );
177
178       }
179
180      });
181
182     }
183
184     index++;
185
186     loadFile( scene, info );
187
188    });
189
190   }
191
192   function getInformation( scene ) {
193
194    apex.server.process( "get_information", {
195
196    }, {
197
198     loadingIndicator : $( "body"),
199     loadingIndicatorPosition : "page",
200     dataType : "text",
201     success : function ( pData ) {
202
203      info = JSON.parse( pData );
204
205      loadFile( scene, info );
206
207     }
208
209    });
210
211   }
212
```

```
213  function init() {
214
215   $( ".openLegend" ).click( function() {
216
217    $( ".tabLegend" ).show( "slow" );
218    $( ".openLegend" ).hide();
219
220    $( ".closeLegend" ).show();
221    $( ".closeLegend" ).css( { "top": "10px", "left": "235px" } );
222
223    $( ".openSwitcher" ).hide();
224    $( ".openHistory" ).hide();
225
226      });
227
228   $( ".closeLegend" ).click(function() {
229
230    $( ".tabLegend" ).hide( "slow" );
231    $( ".closeLegend" ).hide();
232
233    $( ".openLegend" ).show();
234    $( ".openLegend" ).css( { "top": "10px", "left": "10px" } );
235
236    $( ".openSwitcher" ).show();
237    $( ".openSwitcher" ).css( { "top": "10px", "left": "56px" } );
238
239    $( ".openHistory" ).show();
240    $( ".openHistory" ).css( { "top": "10px", "left": "102px" } );
241
242   });
243
244    $( ".openSwitcher" ).click( function() {
245
246    $( ".tabSwitcher" ).show( "slow" );
247    $( ".openSwitcher" ).hide();
248
249    $( ".closeSwitcher" ).show();
250    $( ".closeSwitcher" ).css( { "top": "10px", "left": "235px"} );
251
252    $( ".openLegend" ).hide();
253    $( ".openHistory" ).hide();
254
255   });
256
257   $( ".closeSwitcher" ).click(function() {
258
259    $( ".tabSwitcher" ).hide( "slow" );
260    $( ".closeSwitcher" ).hide();
261
262    $( ".openLegend" ).show();
263    $( ".openLegend" ).css( { "top": "10px", "left": "10px" } );
264
265    $( ".openSwitcher" ).show();
266    $( ".openSwitcher" ).css( { "top": "10px", "left": "56px" } );
267
268    $( ".openHistory" ).show();
269    $( ".openHistory" ).css( { "top": "10px", "left": "102px" } );
270
271   });
272
273   $( ".openHistory" ).click( function() {
274
```

```
275     $( ".tabHistory" ).show( "slow" );
276     $( ".openHistory" ).hide();
277
278     $( ".closeHistory" ).show();
279     $( ".closeHistory" ).css({ "top": "10px", "left": "235px" } );
280
281     $( ".openLegend" ).hide();
282     $( ".openSwitcher" ).hide();
283
284    });
285
286    $( ".closeHistory" ).click(function() {
287
288     $( ".tabHistory" ).hide( "slow" );
289     $( ".closeHistory" ).hide();
290
291     $( ".openLegend" ).show();
292     $( ".openLegend" ).css( { "top": "10px", "left": "10px" } );
293
294     $( ".openSwitcher" ).show();
295     $( ".openSwitcher" ).css( { "top": "10px", "left": "56px" } );
296
297     $( ".openHistory" ).show();
298     $( ".openHistory" ).css( { "top": "10px", "left": "102px" } );
299
300    });
301
302    webglCanvas = document.getElementById( "webgl" );
303
304    renderer = new THREE.WebGLRenderer( { antialias: true } );
305    renderer.setSize( webglCanvas.clientWidth, webglCanvas.clientHeight );
306    renderer.setViewport( 0, 0, webglCanvas.clientWidth, webglCanvas.clientHeight );
307    renderer.setClearColor( 0xFFFFFF );
308
309    renderer.domElement.style.position = "absolute";
310    renderer.domElement.style.zIndex = "1";
311    renderer.domElement.style.top = "0";
312
313    webglCanvas.appendChild( renderer.domElement );
314
315    webglCanvas.addEventListener( "mousedown", onMouseDown );
316    webglCanvas.addEventListener( "resize", onCanvasResize );
317
318    scene = new THREE.Scene();
319
320    camera = new THREE.PerspectiveCamera( 70, webglCanvas.clientWidth / webglCanvas.clientHeight, 0.1,
         100000 );
321    camera.up.set( 0, 0, 1 );
322    camera.position.set( -5, -155, 62 );
323    camera.lookAt( scene.position );
324    scene.add( camera );
325
326    light = new THREE.AmbientLight( 0xFFFFFF, 0.9 );
327    scene.add( light );
328
329    controls = new THREE.OrbitControls( camera );
330    controls.noZoom = false;
331    controls.noRotate = false;
332    controls.noPan = false;
333    controls.zoomSpeed = 1.2;
334    controls.maxZoom = 100000;
335
```

```
336     controls.addEventListener( "change", render );
337
338     getInformation( scene );
339
340   }
341
342   function makeBBox( object ) {
343
344     var bbox = new THREE.Box3().setFromObject( object );
345
346     var xLength = bbox.max.x - bbox.min.x;
347     var yLength = bbox.max.y - bbox.min.y;
348     var zLength = bbox.max.z - bbox.min.z;
349
350     return { bbox, xLength, yLength };
351
352   }
353
354   function createGround( valuesBBox, objectRotation ) {
355
356     var xlength = valuesBBox.xLength;
357     var ylength = valuesBBox.yLength;
358
359     /*var lengthMeters = xlength > ylength ? Math.round( xlength * 0.0254 ) : Math.round( ylength *
          0.0254 );*/
360     var lengthMeters = xlength > ylength ? Math.round( xlength ) : Math.round( ylength );
361
362     /*var centerX = 93815.2265;
363     var centerY = 436409.9599;*/
364     var centerX = ( valuesBBox.bbox.max.x + valuesBBox.bbox.min.x ) / 2;
365     var centerY = ( valuesBBox.bbox.max.y + valuesBBox.bbox.min.y ) / 2;
366
367     var leftX = centerX - lengthMeters;
368     var lowerY = centerY - lengthMeters;
369     var rightX = centerX + lengthMeters;
370     var upperY = centerY + lengthMeters;
371
372     var baseURL = "https://geodata.nationaalgeoregister.nl/top10nlv2/ows?SERVICE=WMS&VERSION=1.1.1&
          REQUEST=GetMap&FORMAT=image/png&TRANSPARENT=true";
373
374     var layers = "terreinvlak,waterdeelvlak,wegdeelvlak";
375     var styles = "top10nlv2:Terrein_vlak_style,top10nlv2:Waterdeel_Vlak_style,top10nlv2:
          Wegdeel_Vlak_style";
376
377     wmsRequest = baseURL + "&STYLES=" + styles + "&LAYERS=" + layers + "&SRS=EPSG:28992&WIDTH=" + 512 + "
          &HEIGHT=" + 512;
378     wmsRequest += "&BBOX=" + leftX + "," + lowerY + "," + rightX + "," + upperY;
379
380     groundTexture = new THREE.TextureLoader().load( wmsRequest );
381     groundMaterial = new THREE.MeshBasicMaterial( { map: groundTexture, transparent: true, opacity: .7 }
          );
382
383     groundLength = xlength > ylength ? xlength * 2 : ylength * 2;
384     groundGeometry = new THREE.PlaneBufferGeometry( groundLength, groundLength );
385
386     ground = new THREE.Mesh( groundGeometry, groundMaterial );
387     ground.position.z = -1 * valuesBBox.bbox.max.z;
388     ground.receiveShadow = true;
389
390     scene.add( ground );
391
392   }
```

```
393
394  function onCanvasResize() {
395
396   renderer.setSize( webglCanvas.clientWidth, webglCanvas.clientHeight );
397   renderer.setViewport( 0, 0, webglCanvas.clientWidth, webglCanvas.clientHeight );
398
399   camera.aspect = ( webglCanvas.clientWidth / webglCanvas.clientHeight );
400   camera.updateProjectionMatrix();
401
402  }
403
404  function onMouseDown( event ) {
405
406   event.preventDefault();
407
408   canvasBounds = renderer.context.canvas.getBoundingClientRect();
409
410   mouse.x = ( ( event.clientX - canvasBounds.left ) / ( canvasBounds.right - canvasBounds.left) ) * 2 -
          1;
411   mouse.y = - ( ( event.clientY - canvasBounds.top ) / ( canvasBounds.bottom - canvasBounds.top ) ) * 2
          + 1;
412
413   vector = new THREE.Vector3( mouse.x, mouse.y, 1 );
414   vector.unproject( camera );
415
416   raycaster = new THREE.Raycaster( camera.position, vector.sub( camera.position ).normalize() );
417   raycaster.setFromCamera( mouse, camera );
418
419   intersects = raycaster.intersectObjects( objects, true );
420
421   if ( $( ".years input:radio" )[1].checked ) {
422
423    if ( intersects.length > 0 ) {
424
425     for ( var i = 0; i < intersects.length; i++ ) {
426
427      $( "#info" ).show( "slow" );
428
429      if ( intersects[i].object != intersected2 ) {
430
431       if ( intersected2 ) intersected2.material = intersected2.currentMaterial2;
432
433       intersected2 = intersects[i].object;
434
435       intersected2.currentMaterial2 = intersects[i].object.material;
436
437       intersected2.material = new THREE.MeshPhongMaterial( { color: 0x1E90FF, transparent: true,
               opacity: .9 } );
438       intersected2.material.needsUpdate = true;
439
440       document.getElementById( "beheerobj" ).value = "Brug (bewegbaar)";
441
442       if ( intersected2.parent.userData.Name == "Aandrijving en bewegingswerk" ) {
443
444         document.getElementById( "discipline" ).value =  "Elektrotechniek";
445
446       } else {
447
448         document.getElementById( "discipline" ).value = "Kunstwerken";
449
450       }
451
```

```
      document.getElementById( "element" ).value = intersected2.parent.userData.Name;
      document.getElementById( "inspdate" ).value = intersected2.parent.userData.InspectionDate07;
      document.getElementById( "inspector" ).value = intersected2.parent.userData.Inspector07;

      showCondition( intersected2.parent.userData.Score12, intersected2.parent.userData.Score07,
           intersected2.parent.userData.Score02 );

     }

    }

   } else {

   $( "#info" ).hide( "slow" );

    if ( intersected2 ) intersected2.material = intersected2.currentMaterial2;

    intersected2.currentMaterial2 = intersects[i].object.material;

    intersected2 = null;

   }

  } else if ( $( ".years input:radio" )[2].checked ) {

   if ( intersects.length > 0 ) {

    for ( var i = 0; i < intersects.length; i++ ) {

     $( "#info" ).show( "slow" );

     if ( intersects[i].object != intersected3 ) {

      if ( intersected3 ) intersected3.material = intersected3.currentMaterial3;

      intersected3 = intersects[i].object;

      intersected3.currentMaterial3 = intersects[i].object.material;

      intersected3.material = new THREE.MeshPhongMaterial( { color: 0x1E90FF, transparent: true,
           opacity: .9 } );
      intersected3.material.needsUpdate = true;

      document.getElementById( "beheerobj" ).value = "Brug (bewegbaar)";

      if ( intersected3.parent.userData.Name == "Aandrijving en bewegingswerk" ) {

       document.getElementById( "discipline" ).value =  "Elektrotechniek";

      } else {

       document.getElementById( "discipline" ).value = "Kunstwerken";

      }

      document.getElementById( "element" ).value = intersected3.parent.userData.Name;
      document.getElementById( "inspdate" ).value = intersected3.parent.userData.InspectionDate02;
      document.getElementById( "inspector" ).value = intersected3.parent.userData.Inspector02;

      showCondition( intersected3.parent.userData.Score12, intersected3.parent.userData.Score07,
           intersected3.parent.userData.Score02 );

```

```
511       }
512
513      }
514
515     } else {
516
517     $( "#info" ).hide( "slow" );
518
519      if ( intersected3 ) intersected3.material = intersected3.currentMaterial3;
520
521      intersected3.currentMaterial3 = intersects[i].object.material;
522
523      intersected3 = null;
524
525     }
526
527    } else {
528
529     if ( intersects.length > 0 ) {
530
531      for ( var i = 0; i < intersects.length; i++ ) {
532
533       $( "#info" ).show( "slow" );
534
535       if ( intersects[i].object != intersected1 ) {
536
537        if ( intersected1 ) intersected1.material = intersected1.currentMaterial1;
538
539        intersected1 = intersects[i].object;
540
541        intersected1.currentMaterial1 = intersects[i].object.material;
542
543        intersected1.material = new THREE.MeshPhongMaterial( { color: 0x1E90FF, transparent: true,
              opacity: .9 } );
544        intersected1.material.needsUpdate = true;
545
546        document.getElementById( "beheerobj" ).value = "Brug (bewegbaar)";
547
548        if ( intersected1.parent.userData.Name == "Aandrijving en bewegingswerk" ) {
549
550          document.getElementById( "discipline" ).value =  "Elektrotechniek";
551
552        } else {
553
554          document.getElementById( "discipline" ).value = "Kunstwerken";
555
556        }
557
558        document.getElementById( "element" ).value = intersected1.parent.userData.Name;
559        document.getElementById( "inspdate" ).value = intersected1.parent.userData.InspectionDate12;
560        document.getElementById( "inspector" ).value = intersected1.parent.userData.Inspector12;
561
562        showCondition( intersected1.parent.userData.Score12, intersected1.parent.userData.Score07,
              intersected1.parent.userData.Score02 );
563
564       }
565
566      }
567
568     } else {
569
570      $( "#info" ).hide( "slow" );
```

```
571
572     if ( intersected1 ) intersected1.material = intersected1.currentMaterial1;
573
574     intersected1.currentMaterial1 = intersects[i].object.material;
575
576     intersected1 = null;
577
578    }
579
580   }
581
582  }
583
584  function animate() {
585
586   window.requestAnimationFrame( animate ) ||
587   window.mozRequestAnimationFrame( animate ) ||
588   window.webkitRequestAnimationFrame( animate ) ||
589   window.msRequestAnimationFrame( animate ) ||
590   window.oRequestAnimationFrame( animate );
591
592   controls.update();
593
594   render();
595
596  }
597
598  function render() {
599
600   camera.lookAt( scene.position );
601   renderer.render( scene, camera );
602
603  }
604
605  previousSel = null;
606
607  $( "#t_Body_details" ).removeClass( "is-collapsed" );
608
609  obsurvTheme.scaleDetails();
610
611 }
612
613 });
614
615 /*function convertToECEF( lat, lon, h ) {
616
617  a = 6378137;
618  f = 1/298.257223563;
619  e = 2 * f - Math.pow( f, 2 );
620
621  N = a / Math.sqrt( 1 - e * Math.pow( Math.sin( THREE.Math.degToRad( lat ) ), 2 ) );
622  X = ( N + 0 ) * Math.cos( THREE.Math.degToRad( lat ) ) * Math.cos( THREE.Math.degToRad( lon ) );
623  Y = ( N + 0 ) * Math.cos( THREE.Math.degToRad( lat ) ) * Math.sin( THREE.Math.degToRad (lon ) );
624  Z = ( ( 1 - e ) * N + 0 ) * Math.sin( THREE.Math.degToRad ( lat ) );
625
626  return { x: X, y: Y, z: Z };
627
628 }*/
629
630 function colorComponent( component, score ) {
631
632  if ( score == 1 ) {
```

```
633
634     component.material = new THREE.MeshPhongMaterial( { color: 0x2FC02F, transparent: true, opacity: .9 }
              );
635     component.material.needsUpdate = true;
636
637   } else if ( score == 2 ) {
638
639     component.material = new THREE.MeshPhongMaterial( { color: 0x9ACD32, transparent: true, opacity: .9 }
              );
640     component.material.needsUpdate = true;
641
642   } else if ( score == 3 ) {
643
644     component.material = new THREE.MeshPhongMaterial( { color: 0xFFFF00, transparent: true, opacity: .9 }
              );
645     component.material.needsUpdate = true;
646
647   } else if ( score == 4 ) {
648
649     component.material = new THREE.MeshPhongMaterial( { color: 0xFF8C00, transparent: true, opacity: .9 }
              );
650     component.material.needsUpdate = true;
651
652   } else if ( score == 5 ) {
653
654     component.material = new THREE.MeshPhongMaterial( { color: 0xFF0000, transparent: true, opacity: .9 }
              );
655     component.material.needsUpdate = true;
656
657   } else {
658
659     component.material = new THREE.MeshPhongMaterial( { color: 0xCC0000, transparent: true, opacity: .9 }
              );
660     component.material.needsUpdate = true;
661
662   }
663
664 }
665
666 function showGround() {
667
668   if ( $( ".toggleSwitch input:checkbox" ).is( ":checked" ) ) {
669
670     ground.visible = true;
671
672   } else {
673
674     ground.visible = false;
675
676   }
677
678 }
679
680 function showObject() {
681
682   for (var i = 0; i < $( ".elements input:checkbox" ).length; i++) {
683
684         if ( $( ".elements input:checkbox" )[i].checked ) {
685
686     objects[i].visible = true;
687
688    } else {
```

```
689
690     objects[i].visible = false;
691
692    }
693
694   }
695
696  }
697
698  function showHistory() {
699
700   for ( var i = 0; i < objects.length; i++ ) {
701
702    objects[i].traverse( function ( child ) {
703
704     if ( child instanceof THREE.Mesh ) {
705
706       if ( $( ".years input:radio" )[1].checked ) {
707
708       colorComponent( child, objects[i].userData.Score07 );
709
710       } else if ( $( ".years input:radio" )[2].checked ) {
711
712      colorComponent( child, objects[i].userData.Score02 );
713
714       } else {
715
716      colorComponent( child, objects[i].userData.Score12 );
717
718       }
719
720     }
721
722    });
723
724   }
725
726  }
727
728  function showCondition( score1, score2, score3 ) {
729
730   colors = { 1: "#2FC02F", 2: "#9ACD32", 3: "#FFDF00", 4: "#FF8C00", 5: "#FF0000", 6: "#CC0000" };
731
732   document.getElementById( "score1" ).style.background = colors[score1];
733   document.getElementById( "score2" ).style.background = colors[score2];
734   document.getElementById( "score3" ).style.background = colors[score3];
735
736  }
737
738  $(document).on('click', '#t_Button_annuleren', function () {
739
740   ...
741   $( ".a-IRR-table" ).find( "tr.is-selected" ).removeClass( "is-selected" );
742   $( "#t_Body_details" ).addClass( "is-collapsed" );
743
744  });
745
746  $(document).on( "apexafterrefresh", "#obs_kunstwerken", function ( dataId ) {
747
748   $( "#obs_kunstwerken tbody tr td span[data-id=" + dataId + "]").parent().parent().click();
749
750  });
```

E PL/SQL CODE

Listing E.1: PL/SQL code for retrieving data from the Oracle database

```
CREATE OR REPLACE PACKAGE BODY KUN_INSPECTIES_APEX AS
 PROCEDURE  GET_INFORMATION
 IS
 EL_ID OBS_ACC.KUN_INSPECTIES_TEST.KWL_ID%TYPE;
 INSPDATE OBS_ACC.KUN_INSPECTIES_TEST.DATUM%TYPE;
 INSPECTOR OBS_ACC.KUN_INSPECTIES_TEST.INSPECTEUR%TYPE;
 SCORE OBS_ACC.KUN_INSPECTIES_TEST.SCORE%TYPE;
 TYPE KUN_REC_TT IS TABLE OF KUN_INSPECTIES_TEST%ROWTYPE;
 KUN_REC_T KUN_REC_TT;
 KUN_IDX PLS_INTEGER;
 BEGIN
  SELECT * BULK COLLECTION INTO KUN_REC_T FROM KUN_INSPECTIE_TEST;
  KUN_IDX := KUN_REC_T.FIRST;
  HTP.P('[');
  WHILE KUN_IDX IS NOT NULL
  LOOP
  IF KUN_IDX != KUN_REC_T.FIRST
  THEN
  HTP.P(',');
  END IF;
  HTP.P('{ "ElementID:"'}||KUN_REC_T(KUN_IDX).KWL_ID||',
            "InspDate:"'||KUN_REC_T(KUN_IDX).DATUM||'",
            "Inspector:"'||KUN_REC_T(KUN_IDX).INSPECTEUR||'",
            "Score:"'||KUN_REC_T(KUN_IDX).SCORE||' }');
  KUN_IDX := KUN_REC_T.NEXT(KUN_IDX);
  END LOOP;
  HTP.P(']');
 END GET_INFORMATION;
END KUN_INSPECTIES_APEX;
```

F | QUESTIONNAIRE SURVEY

Integration of 3D BIM Models in a Web GIS for Supporting Decision Making in Life Cycle Asset Management | Prototype Evaluation

Welcome to the questionnaire! This survey aims to assess the usability of a prototype built during a graduation project at Delft University of Technology. The participation of experts in diverse fields will help to indicate any issues related to the prototype development and operation, and get ideas for its improvement.

The research focuses on the integration of 3D BIM models in OBSURV in order to support decision making in Life Cycle Asset Management. Various functional and non-functional requirements have been specified at the beginning of the project for finding a suitable approach to build the prototype.

Please try to answer all the questions to allow for better prototype evaluation. The questionnaire will take 5 to 10 minutes.

NEXT

Never submit passwords through Google Forms.

Integration of 3D BIM Models in a Web GIS for Supporting Decision Making in Life Cycle Asset Management | Prototype Evaluation

*Required

Personal Information

Name:

Your answer

Surname:

Your answer

Occupation: *

Your answer

Are you familiar with OBSURV? *

- ○ Yes, I am very familiar with OBSURV.
- ○ Yes, I have used OBSURV several times.
- ○ Yes, I have heard of OBSURV, but I have never used it before.
- ○ No, I am not familiar with OBSURV.

BACK NEXT

Never submit passwords through Google Forms.

Integration of 3D BIM Models in a Web GIS for Supporting Decision Making in Life Cycle Asset Management | Prototype Evaluation

*Required

Usability Testing

Do you think navigating around the 3D viewer is easy?

- ○ Yes
- ○ No

How would you grade the following functionalities of the 3D viewer based on their usability? *

	Poor	Fair	Good	Very Good	Excellent
Legend	○	○	○	○	○
Layer Switcher	○	○	○	○	○
Condition Visualization	○	○	○	○	○
Change Detection	○	○	○	○	○
Object Selection	○	○	○	○	○
Information Tab	○	○	○	○	○
Base Map	○	○	○	○	○
Animation	○	○	○	○	○

What is the most useful functionality of the 3D prototype? *

- ○ Legend
- ○ Layer Switcher
- ○ Condition Visualization
- ○ Change Detection
- ○ Object Selection
- ○ Information Tab
- ○ Base Map
- ○ Animation
- ○ Other: _____

What is the main usability issue of the 3D prototype? *

Your answer

BACK NEXT

Never submit passwords through Google Forms.

Integration of 3D BIM Models in a Web GIS for Supporting Decision Making in Life Cycle Asset Management | Prototype Evaluation

Further Improvements

Which other functionalities could be added to the 3D viewer for improving the 3D prototype?

Your answer

Do you have any additional comments / remarks?

Your answer

BACK NEXT

Never submit passwords through Google Forms.

Integration of 3D BIM Models in a Web GIS for Supporting Decision Making in Life Cycle Asset Management | Prototype Evaluation

Thank you for your participation!

Thank you for taking your time to complete this survey! Your contribution is essential for the evaluation of the prototype and the implementation of the 3D viewer in OBSURV in the near future.

BACK SUBMIT

Never submit passwords through Google Forms.

G | REFLECTION

The graduation project was made in cooperation with Delft University of Technology and Sweco Netherlands, and officially started in November 2017. The thesis gave me the opportunity to research a very actual and challenging topic in the Geomatics field, namely the BIM-GIS integration. It was interesting for me to acquire specialised knowledge and experience with BIM design and modelling techniques, and Web GIS development.

The research combines various subjects covered in different courses of the Master's programme in Geomatics. The *3D Modelling* course gave a solid basis for the construction of 3D models and their components, and introduced the different 3D file formats. The knowledge gained in the course was used for the decomposition of BIM models. The *Geo Database Management Systems* course focused on the database management and data manipulation techniques, which was necessary for the data retrieval process within the scope of the research.

The *GIS and Cartography* course introduced GIS as a tool for solving real-world problems and explained the basic concepts of cartography and different reference systems and projections. In addition, the *Positioning and Location Awareness* course extended the knowledge about the coordinate systems by providing practical experience in performing transformations. This gave valuable insights for the georeferencing of BIM models.

The *Geo Web Technology* course was the most essential course for this project, as it provided the basics of web mapping and front-end development with HTML, CSS, and JavaScript. As the 3D prototype is created based on acwebgl, a prior experience in front-end programming was desirable.

Throughout the course of the project, I was able to improve my web development skills by extensively using HTML, CSS, and JavaScript, and learn PL/SQL for data manipulation in Oracle.

Besides the improvement of my technical skills, the research helped me to develop my soft skills as well. In order to gather the requirements for the design of the 3D prototype, I needed to find several AM experts both in the Municipality of Rotterdam and Sweco and conduct on-site interviews with them. This improved my communication skills to a large extent and allowed me to better understand the work procedures in public space management.

During my graduation project in the company, I was positively surprised by the kindness of my colleagues and their willingness to help me with any difficulties. I got continuous

support by several people from the OBSURV Software team when I was developing the 3D prototype.

COLOPHON

This document was typeset using LaTeX. The document layout was generated using the `arsclassica` package by Lorenzo Pantieri, which is an adaption of the original `classicthesis` package from André Miede.


TU Delft
SWECO