International Journal of Geo-Information

MDPI

Article

Prototype of the 3D Cadastral System Based on a NoSQL Database and a JavaScript Visualization Application

Nenad Višnjevac, Rajica Mihajlović, Mladen Šoškić, Željko Cvijetinović and Branislav Bajat *

University of Belgrade, Faculty of Civil Engineering, Bulevar kralja Aleksandra 73, 11000 Belgrade, Serbia; nvisnjevac@grf.bg.ac.rs (N.V.); rajica@grf.bg.ac.rs (R.M.); mladens@grf.bg.ac.rs (M.Š.); zeljkoc@grf.bg.ac.rs (Ž.C.)
* Correspondence: bajat@grf.bg.ac.rs; Tel.: +381-11-3218-579

Received: 3 April 2019; Accepted: 8 May 2019; Published: 10 May 2019

Abstract: 3D cadastral systems are more complex than traditional cadastral systems and they require more complex technical solutions and innovative use of developing technologies. Regarding data integrity and data consistency, 3D cadastral data should be maintained by a Database Management System (DBMS). Furthermore, there are still challenges regarding visualization of 3D cadastral data. A prototype of the 3D cadastral system based on a NoSQL database and a JavaScript application for 3D visualization is designed and tested in order to investigate the possibilities of using new technical solutions. It is assumed that this approach, with further development, could be a good basis for the development of a modern 3D cadastral system. MongoDB database is used for storing data and Cesium JavaScript library is used for 3D visualization. The system uses an LADM (Land Administration Domain Model) based data model. Additionally, script languages, libraries, application programming interfaces (APIs), software and data formats are used for the system development. The case study is based on the real cadastral data. The underground object and building units located below and above the ground level are used to test the proposed data model and the system's functionality. The proposed system needs further development in order to provide full support to a modern 3D cadastral system. However, it allows maintenance of 3D cadastral data and basic 3D visualization with the interactive approach.

Keywords: 3D cadastre; LADM; NoSQL; JavaScript; visualization

1. Introduction

During the past two decades, research on 3D cadastres have been performed [1,2], raising worldwide awareness for the need of 3D cadastral systems. 3D registration of real properties, unequivocal registration of complex 3D property situations and their visualization are necessary in a 3D cadastre to increase transparency in land administration and property registration.

Following the proposed models [3–6] and international standard [7–9], it can be concluded that 3D cadastral systems are more complex than traditional cadastral systems. In other words, 3D cadastral systems require more complex technical solutions. Besides the legal and organizational aspect, additional attention should be dedicated to data models, data storage and visualization of 3D cadastral data. These technical aspects will enable full implementation and usability of a 3D cadastral system. To address these challenges, the study contains a research effort to develop a 3D cadastre prototype.

Since cadastral systems are designed to store property boundaries and right holder information [10], a 3D cadastral system also has to guarantee data integrity and data consistency, for both 3D spatial and textual data. Therefore, it is very important that 3D cadastral data are maintained by a Database Management System (DBMS) [11] because other approaches like file-based systems cannot meet this

requirement. Regarding modeling and storing spatial data, the topological consistency of 3D cadastral data is also important. This implies topological models or rule-based system validated by using well-understood methodologies [12,13]. However, a lot of obstacles to meet all the requirements of a 3D cadastral system still exist, despite the various existing technologies for storing and visualization of 3D data. Relational Database Management Systems (RDBMS) do not natively support 3D topology [14,15] and there are still a lot of challenges in 3D cadastral data visualization [16–18].

New developments in the field of DBMS (such as NoSQL databases) and innovative 3D modeling and visualization technologies are available, now. Additionally, there are other modern technologies, such as Spatial MapReduce systems (SpatialHadoop or GeoSpark) that can be applied for processing spatial objects in an effective way [19,20].

For these reasons, the main purpose of this study is to investigate an alternative way, and innovative use of the developing technologies in DBMS field for 3D cadastre. There is no intention to substitute or decline the use of traditional technologies such as relational databases. In that context, the paper explores the possibilities of implementing a 3D cadastral prototype based on NoSQL database and JavaScript application for 3D visualization. The starting assumption is that such an approach, with further development, could be a good basis for the development of a modern 3D cadastral system. The study also includes research on the advantages and disadvantages of the proposed prototype and what needs to be done to improve such a system towards a fully operational 3D cadastral system.

This study has focused on developing a 3D cadastre prototype facilitating real properties registration with 3D data, enabling improvement of land administration and management in the way that rights on complex 3D situations are secured and disputes minimized. The prototype provides user interface and methods for visualizing 3D spatial objects and it also ensures data integrity and consistency by storing the data within a DBMS. The prototype can be used as a basis for the development of a national 3D digital cadastre system which will increase transparency in land administration and property registration.

More specifically, a LADM (Land Administration Domain Model) based data model is implemented into the physical model based on MongoDB (which is a NoSQL database) and Cesium JavaScript library. MongoDB is selected because it is one of the most popular NoSQL databases and because it has spatial operators so future development on a spatial component could be expected. However, MongoDB currently does not improve the situation of handling 3D topology. In this study, only queries on attributes defined by the LADM based data model are performed in MongoDB. No evaluation is done on spatial queries at all. Cesium library is an open-source JavaScript library and widely used tool for 3D mapping of geospatial data. The prototype is based on free and open-source software and it is expected that it could be used by both professional and non-professional users.

The rest of this paper is organized as follows: Section 2 provides the used methodology. It presents the LADM based data model and introduces the architecture and implementation of the 3D cadastral system. Used technology and data preparation are explained as well. Section 3 contains the case study with real cadastral data showing how the system works. It provides a walk-through example of using the system. Section 4 presents a discussion on the advantages and disadvantages of the system. It also provides conclusions and proposals for further research.

2. Methodology

In order to introduce the proposed 3D cadastral system, three main aspects were defined: (a) a data model, (b) a utilized database i.e., MongoDB, and (c) a 3D visualization approach based on the Cesium JavaScript library. Data source descriptions and data preparation for import into MongoDB were also discussed.

Some other methods and technologies were also applied, particularly the JSONP (JavaScript Object Notation) method and the REST (Representational State Transfer) API service. JSONP is a method for sending JSON (JavaScript Object Notation) objects without cross-domain issues. REST API

is a service that provides access to data stored in MongoDB from client applications. Figure 1 illustrates the architecture of the 3D cadastral system based on the MongoDB database and Cesium library.

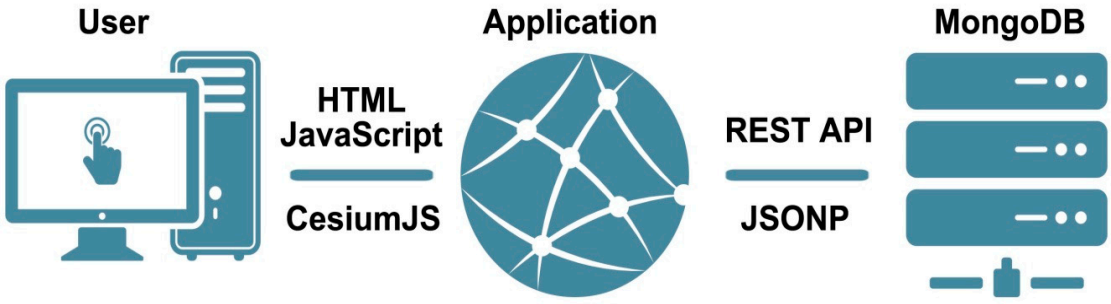

Figure 1. Architecture of the 3D cadastral system based on the MongoDB database and Cesium library.

Table 1 shows the script languages, libraries, APIs, software and data formats used for development of the 3D cadastral system or data preparation. The table contains a description of purpose of use for each technology.

Table 1. Languages, libraries, APIs, software and data formats used for the development of the 3D cadastral system.

Category	Technology	Purpose of Use
Script languages	HTML	The standard markup language used for creating the web page of the 3D cadastral system.
	CSS	A language used to describe the style of the web page.
	JavaScript	A language used to program actions and listening events. It is also used for reading JSON objects and preparing data as input for the Cesium JavaScript library.
	R	Used for preparing JSON objects and multi-surface geometry based on 2D floor plans and heights [21].
Libraries and APIs	Cesium	JavaScript library that enables visualization of 3D content within a web browser [22].
	REST API	Service used for access to data stored in MongoDB from the user application.
	JSONP	A method used for sending JSON data without cross-domain issues.
Software	MongoDB	NoSQL database used for storing, maintaining and processing of the 3D cadastral data [23].
	NoSQL Viewer	Software used for viewing the created database and collections [24].
	QGIS	GIS software used for preparation 2D floor plans, as input for creating 3D multi-surface geometry [25].
Data formats	JSON	The text-based data format used for preparing and sending data.
	BSON	Binary JSON as exists when imported into the MongoDB database.
	Shapefile	The data format used for storing 2D floor plans.

From Table 1, it can be seen that some of the well-known technologies, such as HTML, CSS, and JavaScript are used for the development of the web-based 3D cadastral system. They will not be discussed further within this paper. Additionally, R language and QGIS software are used for data preparation, i.e., converting 2D floor plans to 3D multi-surface geometry.

All these technologies were used to design and implement an appropriate service chain in order to meet the needs of a 3D cadastral system and the main aspects are explained in the following subsections.

2.1. Data Model

Since the case study is based on the cadastral data from Serbia (see Section 3), the proposed prototype of the 3D cadastral system is based on the 3D real estate cadastre data model for the Republic of Serbia (Figure 2).

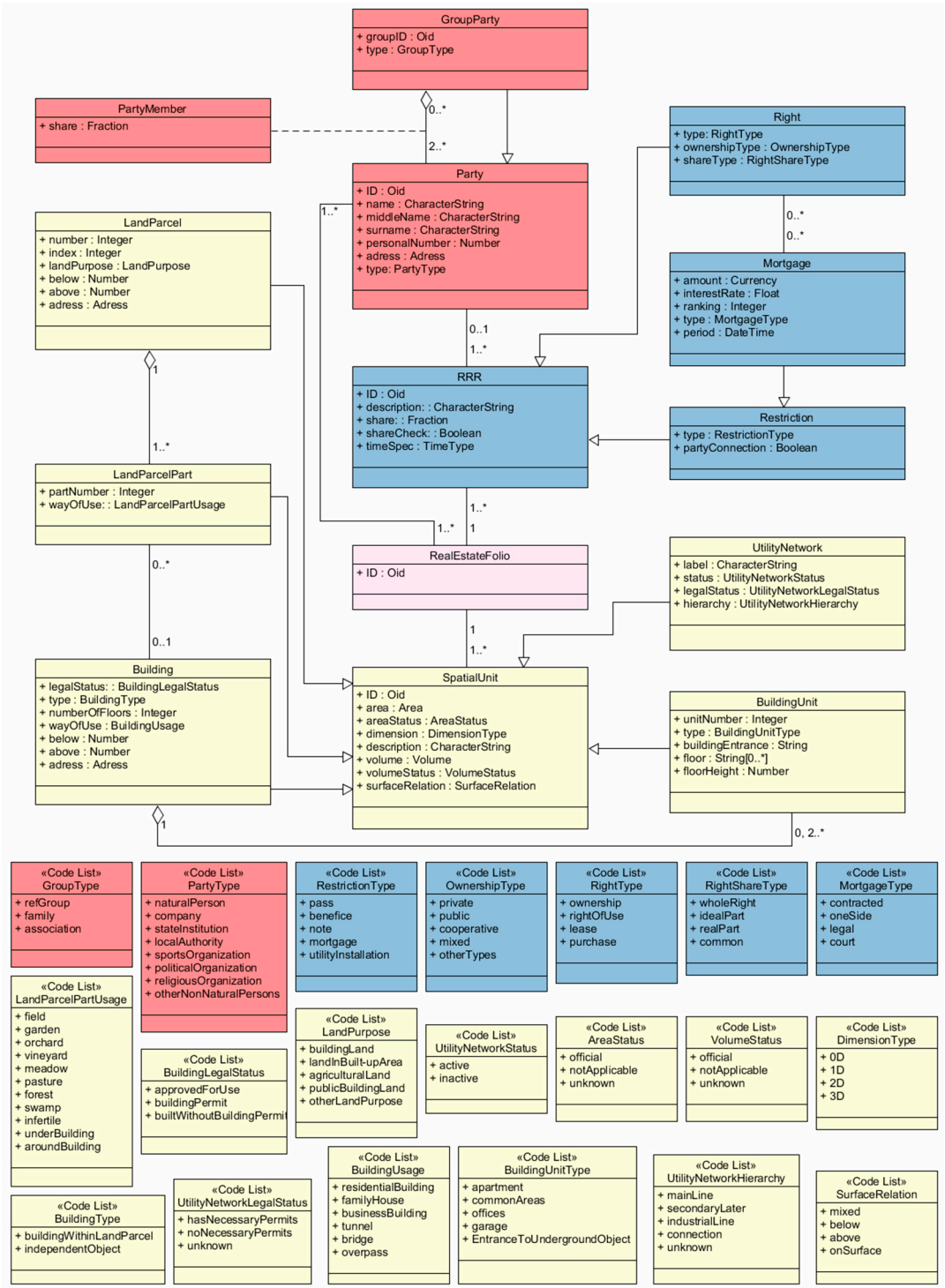

Figure 2. The main classes of the 3D real estate cadastre data model.

The data model is LADM based and it was developed using the results of previous research on the developing Serbian 3D cadastral system [26]. An important characteristic of the data model is that spatial data, e.g., 3D geometry representation is obtained using multi-surface geometry [27], as proposed by LADM.

Figure 2 provides an overview of the main classes of the data model. The classes are organized in groups (packages of LADM standard), i.e., Party Package (red), Administrative Package (blue) and Spatial Unit Package (yellow). In the case study, some data model classes are implemented into a physical model as MongoDB collections (see Section 2.2).

Figure 3 shows the classes of the spatial component. SpatialUnit class represents the conjunction point between several data model components presented in Figures 2 and 3. The 2D and 3D representations of spatial units are possible by using boundary face strings and boundary faces. The fact that the prototype of the 3D cadastral system uses a data model based on the LADM standard enables its modification and use in other countries with 3D cadastral models that are also based on the LADM standard.

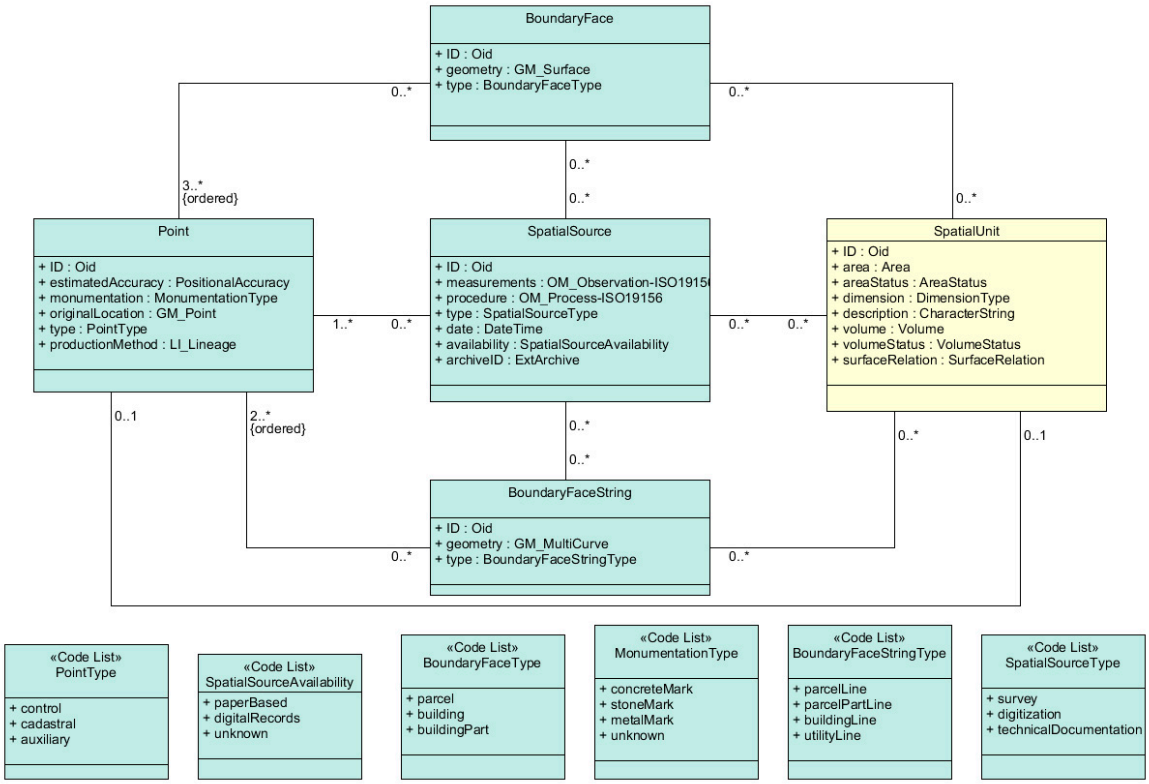

Figure 3. The spatial component of the data model.

2.2. MongoDB

Within this study, the MongoDB database was used for storing, maintenance and processing of the real cadastral data. MongoDB is free, open-source, and one of the most popular NoSQL databases and the fifth most popular DBMS system [28]. The installation process on a Linux server is simple and instructions for various Linux systems (including other operating systems) can be found at the official MongoDB manual website [23].

The first step in the implementation of the prototype 3D cadastral system was to create a database and collections. Collections can be viewed as tables of a relational database and can be defined as a set of documents grouped together. A document is a record in a collection and it is also a basic data unit in MongoDB. Documents can be prepared and viewed as JSON objects but when imported into the database they exist as BSON (Binary JSON) format, which is the binary representation of a JSON document [23].

For this study, the cadastral data were prepared as JSON documents with the schema defined by the proposed data model (Figure 2). The example of the prepared JSON document for a land parcel is as follows:


```
{
"LandParcel":{
"Number":5352,
"Index":2,
"LandPurpose":1,
"Below":"",
"Above":"",
"Adress":201316,
"Area":1339,
"AreaStatus":1,
"Dimension":3,
"Description":"",
"Volume":"",
"VolumeStatus":2,
"SurfaceRelation":4,
"SpatialSourceID":742,
"RealEstateFolioID":9568,
"Geometry":"5baa7d588d32f1224c7c71bd"
}}
```


The example of geometry encoding (for the same land parcel) is as follows:


```
{"BoundaryFaceString":{
"Geometry":"7444480.63 4972434.28 80.8 7444499.97 4972463.49 80.62 7444488.31813887
4972471.51517706 80.4 7444486.20115581 4972472.96421546 80.35 7444480.9399997
4972476.61000021 80.62 7444478.27999999 4972478.45 80.56 7444468.83 4972484.27 80.37
7444448.89 4972454.11 80.6 7444461.12999726 4972446.46000171 80.66",
"Type":"1"
}}
```


The prepared JSON documents were then imported into the database. It is possible to load one or more JSON documents in one step. The following code creates a database, collection and imports JSON document.

Algorithm 1: Code for creating a database, collection and importing a JSON document

```
//If the database exists it calls and connects to it. Else the new one is created.
mongo 3DCadastre
//Imports the JSON document into the collection. If the collection does not exist it creates the new one on the fly.
mongoimport –db 3DCadastre –collection LandParcels –type json –file ./Data/2054.json –jsonArray
//Imports many JSON documents at one step.
for i in ./Data/*.json; do mongoimport –db 3DCadastre –collection LandParcels –type json –file $i –jsonArray;
done
```

Data stored in the MongoDB database can be queried and updated using queries similar to ones used within relational databases. This following code provides an example of queries using the mongo shell.

Algorithm 2: Code for querying and updating the data in the database

```
//Query for getting all the data for specified land parcel.
//This query corresponds to the following SQL code:
//SELECT * FROM LandParcels WHERE Number = 1476 AND Index = 2
use 3DCadastre
db.LandParcels.find({"LandParcel.Number":1476, "LandParcel.Index":2})
//Query for getting the certain attributes of the land parcel.
//This query corresponds to the following SQL code:
//SELECT RealEstateFolioID FROM LandParcels WHERE Number = 1476 AND Index = 2
use 3DCadastre
db.LandParcels.find({"LandParcel.Number":1476,"LandParcel.Index":2},{_id:0,"LandParcel.RealEstateFolioID":1})
//Query for updating the certain attributes of the land parcel.
//This query corresponds to the following SQL code:
//UPDATE LandParcels SET RealEstateFolioID = 244 WHERE Number = 1476 AND Index = 2
use 3DCadastre
db.LandParcels.update({"LandParcel.Number":1476,"LandParcel.Index":2},
{$set:{"LandParcel.RealEstateFolioID":244}})
```

Other examples of selecting and updating data can be found at the MongoDB manual website [23] and in the previous research [29].

2.3. Cesium JavaScript Library

Cesium is an open source JavaScript library that enables the visualization of 3D content within a web browser [22]. Within this study, it was used for visualization of 3D geometries made up of multi-surfaces (as defined by the data model, see Section 2.1) that are stored in the MongoDB database. Besides the visualization of 3D geometry, it was also used for providing textual cadastral data and as a platform for querying data. Cesium library downloads and the guide for quick setting and running the library can be found at the official Cesium website [22].

The main step in the implementation of the Cesium library for the purpose of visualization of 3D cadastral data is to read 3D cadastral data provided by MongoDB queries. Since MongoDB, by using REST API and JSONP methods, provides a JSON object, it means that it is necessary to read the JSON data and convert the results into a form readable by the Cesium library. Once it is converted into a Cesium readable form, all visualizations are done by the Cesium library itself.

Cesium JavaScript library provides different levels of graphical elements. There are Primitive API and Entity API. Primitives are lower level elements and they provide better control of the 3D geometry visualization. For example, it is possible to display a geometry based on coordinates. Primitive API is used for visualization of 3D cadastral data based on an array of multi-surface coordinates.

In this study, the cadastral data were read from the MongoDB database as JSON documents. Multi-surface coordinates were converted to an array and then provided to the Cesium Primitive API for visualization. An example of the JavaScript code that provides coordinates of a building unit to Primitive API is as follows:

Algorithm 3: Code for visualization of a building unit by using Cesium Primitive API and an array of multi-surface coordinates

```
//BuildingUnitCoordinates is an array that contains building unit coordinates
var BuildingUnitCoordinates =[];
var j;
var mypositions = Cesium.Cartesian3.fromDegreesArrayHeights(BuildingUnitCoordinates);
var numPositions = mypositions.length;

var pos =new Float64Array(numPositions *3);
var normals =new Float32Array(numPositions *3);
for(var i =0; i < numPositions;++i){
pos[ i*3]= mypositions[i].x;
pos[i *3+1]= mypositions[i].y;
pos[i *3+2]= mypositions[i].z;}

var geometry =new Cesium.Geometry({vertexFormat : Cesium.VertexFormat.ALL,
attributes:{ position:new Cesium.GeometryAttribute({
componentDatatype: Cesium.ComponentDatatype.DOUBLE,
componentsPerAttribute:3,
values: pos})},
indices:new Uint32Array(Array.apply(null,{length:(numPositions)}).map(Number.call,Number)),
primitiveType: Cesium.PrimitiveType.TRIANGLES,boundingSphere: Cesium.
BoundingSphere.fromVertices(pos)});
Cesium.GeometryPipeline.computeNormal(geometry);

var BuildingUnitInstance =new Cesium.GeometryInstance({
geometry: geometry,
attributes:{
color:new Cesium.ColorGeometryInstanceAttribute.fromColor(Cesium.Color.fromCssColorString("#ffff99"))},
show:new Cesium.ShowGeometryInstanceAttribute(true), id :'Building unit'});

var BuildingUnit = viewer.scene.primitives.add(new Cesium.Primitive({
geometryInstances:[BuildingUnitInstance],
asynchronous:false,
appearance:new Cesium.PerInstanceColorAppearance({closed:true, translucent:false})}));
```

There are several studies [29–34] that use Cesium library as a platform for visualization of 3D cadastral data, smart city applications, BIMs (Building Information Modeling) and other spatial data.

Figure 4 provides an example of multi-surface geometry visualized by Cesium JavaScript library.

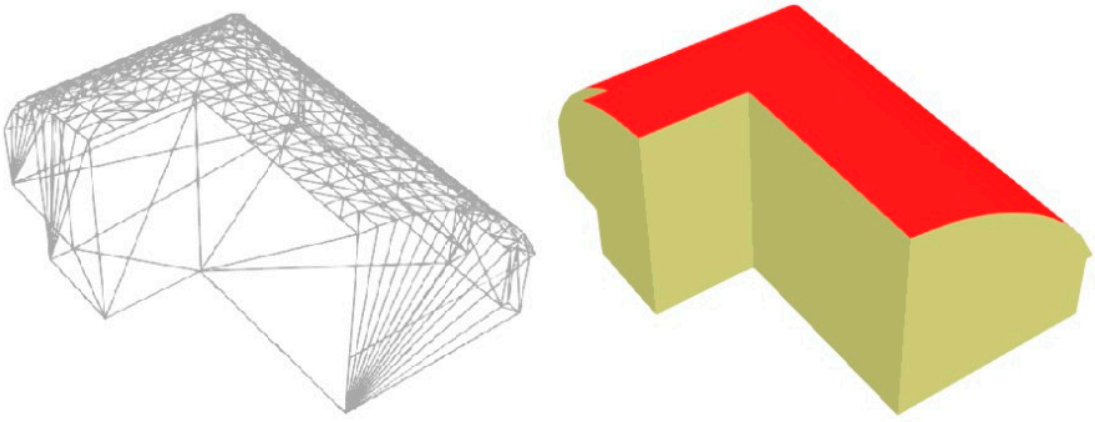

Figure 4. Geometry visualized by Cesium library.

2.4. Data Preparation

The case study 3D cadastral data were created using existing data sources such as drawings of floor plans (Figure 5) and cross-sections. These drawings were provided by a licensed surveying agency. Other publicly available data sources were also used. A Digital Platform of National Spatial Data Infrastructure (GeoSrbija) [35] was used as a source for land parcels data. A digital real estate cadastre database (web application eKatastar) [36] was used to get other textual data on real estate objects.

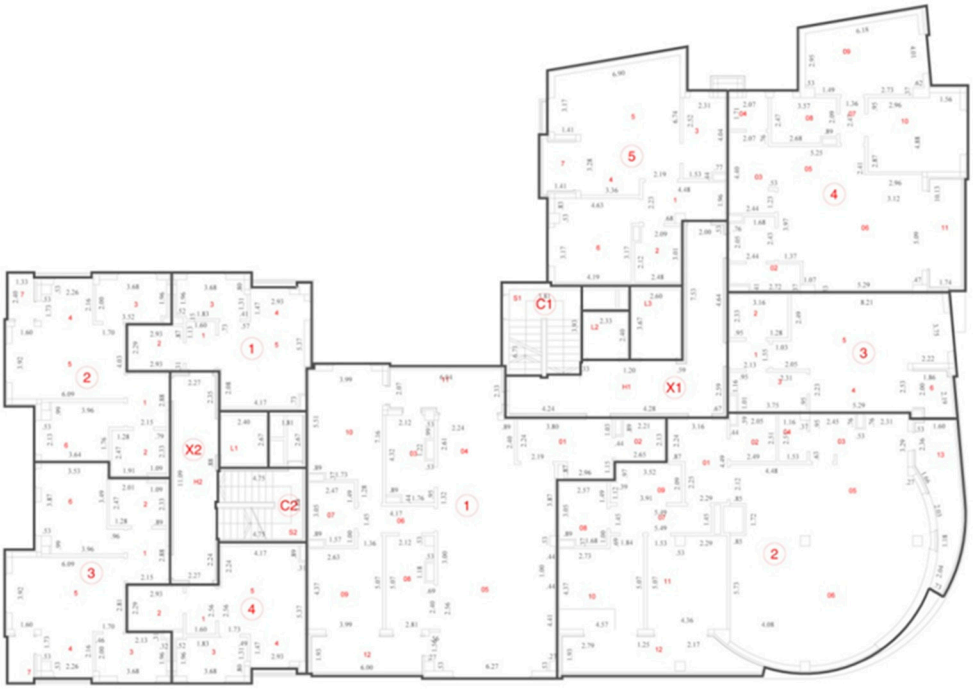

Figure 5. Example of the floor plan used for creating 3D cadastral data.

Manual work is required to create 3D geometry (multi-surfaces that form a 3D object) since there are still no fully developed procedures that will provide 3D geometry automatically. It is necessary to reconstruct 3D geometry of a objects by using available data sources and GIS or CAD software tools. However, many objects (building units in the case study, see Section 3) have simple geometry and can be described using base polygon (footprint) and height, which enables creating 3D geometry automatically. In this study, scripts in R language were written and used for the automatic creation of multi-surfaces that form a 3D geometry. Input for these scripts are 2D polygons as the building unit's footprint and its height. The output is a set of surfaces in JSON format with the structure defined by the used data model. Multi-surfaces were created manually using appropriate CAD software for other objects that are more complex.

3. The Implementation

The case study consists of real estate objects in two separate locations of Belgrade. At the first location, a complex building located on the corner of the streets has to be processed and stored into the 3D cadastre. The building includes residential, business and common property units. In addition, there are building units located below and above the ground level. An underground shelter needs to be processed at the second location. The underground shelter includes an entrance located above the surface of the terrain, underground structure and two ventilation openings located both above and below the surface of the terrain.

Figure 6 provides photos of the case study building. Table 2 provides the main descriptive data on the building that was used for registration and visualization within the case study.

Figure 6. Study building.

Table 2. Description of the case study building.

Category	Value	Description
Location	44°48′17.2″N 20°29′05.9″E	The building is located in Belgrade, at the corner of Dimitrija Tucovića street and Igmanska street.
The highest point of the building	altitude 147.60 m	The highest point of the building is located 31.75 m above the ground level.
The lowest point of the building	altitude 108.10 m	The lowest point of the building is located 7.75 m below the ground level.
Number of floors	12	There are three underground floors, ground floor and eight floors above the ground level.
Number of building units	96	Besides apartments and offices, it includes common parts such as entrance hall, hall, staircase, elevator, warehouse, entrance to the garage, etc.

As previously explained (Section 2.4), existing data sources, such as floor plans and cross-sections, were used for creating multi-surfaces of building units. In this particular case, there were enough existing data to create 3D geometries of building units. This case study also includes land parcels and heights at border points. Data on land parcels were also created based on existing data sources and web services [35,36].

Figure 7 provides photos of the underground shelter. Table 3 provides the main descriptive data on the underground shelter.

The existing cadastral data and surveying records provided by a licensed surveying agency were used as data sources for creating 3D cadastral data on the underground shelter. The available data were almost enough to create the multi-surface 3D geometry for the underground shelter, as defined by the data model. Additional measurements were required to define the geometry of ventilation openings. The lowest point of the underground shelter was estimated since there was no available documentation and it was not possible to measure it. Web services for providing cadastral data were also used as a data source for land parcel data.

As described in the methodology section (Section 2), the 3D cadastral data on building units, underground shelter, and land parcels are prepared as JSON documents that were imported into MongoDB and visualized by the application based on Cesium JavaScript library.

Figure 7. Study underground shelter.

Table 3. Description of the case study underground shelter.

Category	Value	Description
Location	44°53′40.3″N 20°17′29.5″E	The underground shelter is located in Belgrade, in the Svetog Serafima Sarovskog street.
The highest point of the underground shelter	altitude 83.4 m	The highest point of the underground shelter is located on the top of the entrance. The highest point of the underground structure is located 1.6 m below the highest terrain point.
The lowest point of the underground shelter	altitude 77.3 m	The lowest point of the underground shelter is estimated at 5.4 m below the highest terrain point.
The highest terrain point	altitude 82.7 m	The highest terrain point is located above the middle of the underground shelter structure.
Number of terrain points	43	The number of heights points used to present the terrain model above the underground shelter.
Number of units	1	The underground shelter is observed as a one-part object that includes entrance, underground structure and two ventilation openings.

The user interface of the software application for 3D visualization of cadastral data is divided into several sections (Figure 8).

The main section is the map canvas section where all 3D cadastral data are visualized. Unlike most of the Cesium based applications, it is noticeable that this application does not have a virtual globe as a base layer. The reason for turning off the virtual globe feature is for better visualization of 3D objects located below the ground level. Without the virtual globe, it is possible to see 3D objects from all sides (see Figure 9). Land parcels and height points provide the contexts of what is below and what is above the ground level. The virtual globe can be turned on by using the special button in the control section of the user interface.

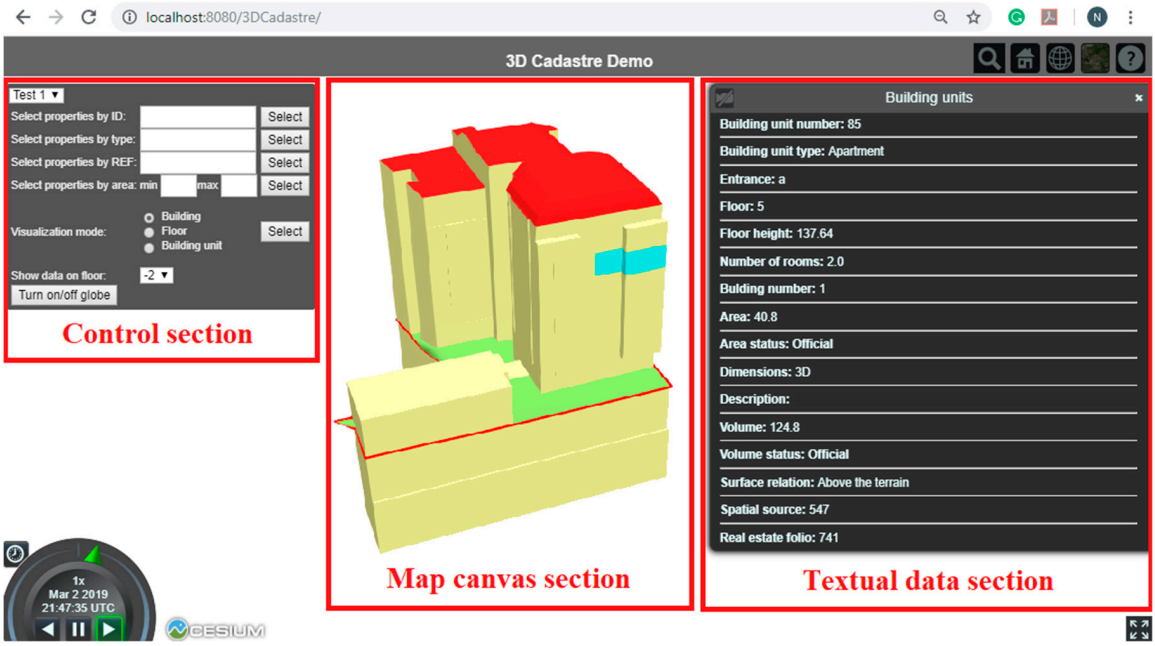

Figure 8. User interface of the 3D cadastre application (available online: http://osgl.grf.bg.ac.rs/3dcad/).

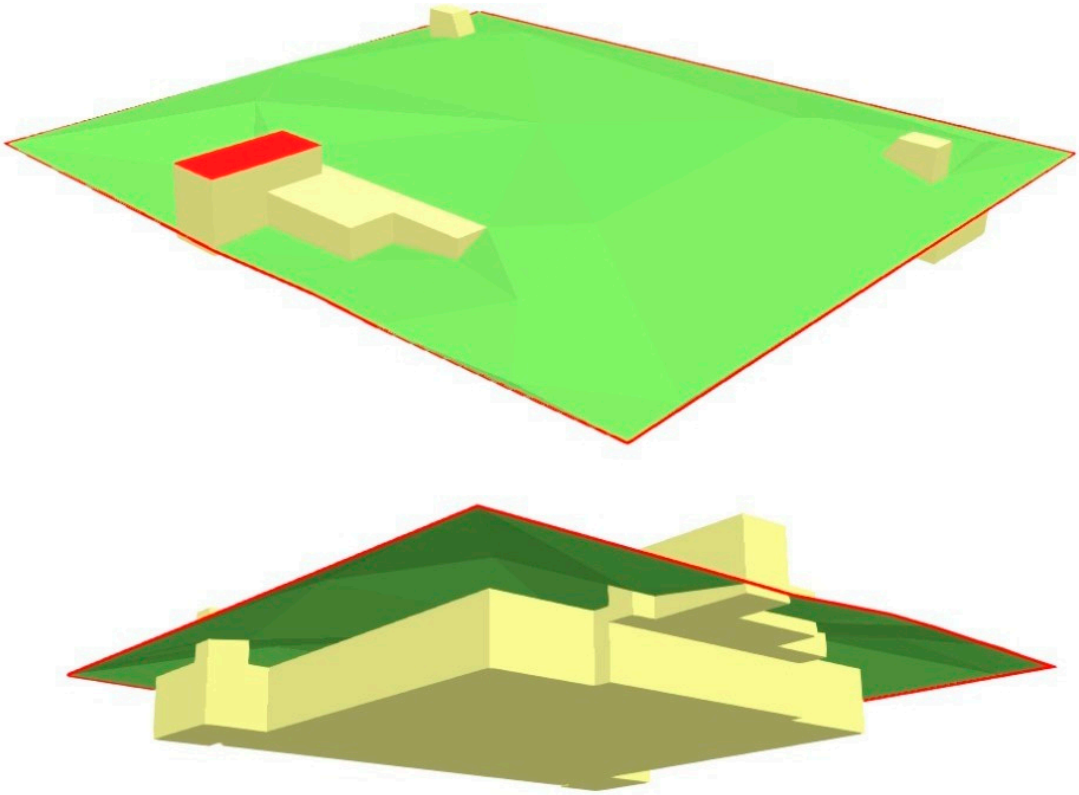

Figure 9. The underground shelter viewed above and below the surface of the terrain.

The main section is also interactive. It means that a user can rotate, pan, zoom and click the content. For example, a user can click on a building unit or other object, which activates the textual data section displaying textual data on a selected item.

The control section of the user interface provides different options for controlling the application and querying data. Beside the button for turning on/off the virtual globe, there are buttons for building unit mode, floor mode or building mode visualization. The building unit mode shows 3D geometry

only for a selected building unit (see Figure 4). In that way, a selected building unit can be viewed from all sides. The floor mode enables viewing of building units located on the selected floor (Figure 10). The building mode is the default option (Figure 8). It shows all building units or underground shelter and enables the selection of units by the mouse pointer.

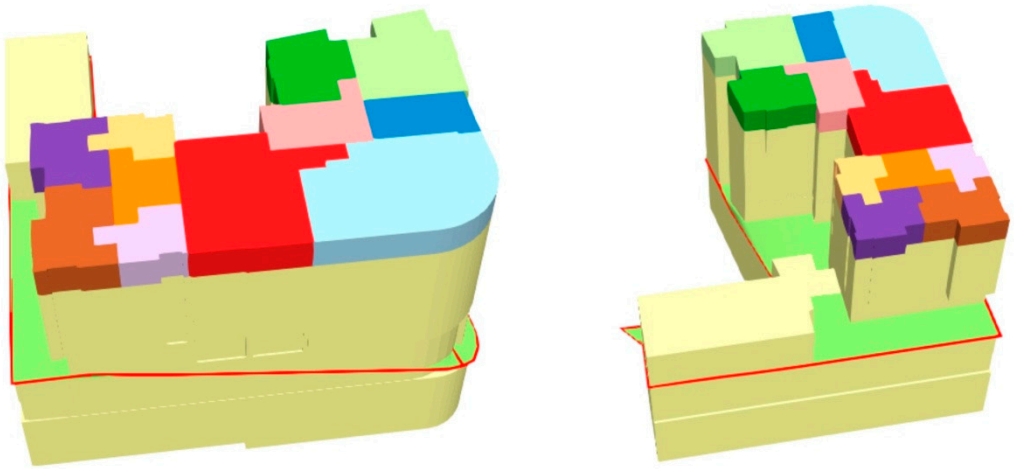

Figure 10. Visualization by using the floor mode.

Figure 10 shows an example of visualization using the floor mode. The building units on the selected floor are rendered in different colors.

By using options in the control section, it is possible to query data by unit ID, unit type, area, number of real estate folio, etc. It is possible to extend the control section in order to implement a query for any textual or numeric attribute defined by the data model. Data filtering could be implemented using a query language as presented in the methodology section since the data are stored within a MongoDB database. In that case, the defined data filtering is done by the database, and the visualization process in Cesium would be the same. Selections are done on the application side using JavaScript language. Selected units are colored at the map canvas section after running a query.

Figure 11 provides the results of different queries that were run using options in the control section.

Selection of the objects that are below or above the ground surface is still possible using an attribute defined by the data model, without using spatial query. The current lack of the developed application is that there are no options for spatial queries such as selection of neighboring objects. This is not supported by the current spatial operators in the MongoDB database. The proposed application also does not support topology rules. In other words, there are no options to check if there are intersections or gaps between building units. As described in previous research [29], it is possible to implement some of the topology rules using CSG (Constructive Solid Geometry) JavaScript library [37]. However, this application would only work on simple objects. These options should be further developed since validation of cadastral data is a very important issue due to its legal repercussions. Certainly, situations such as overlapping properties are not allowed. MongoDB supports 2D spatial index which can be useful for efficient handling of large datasets. The dataset for this case study is rather small, and therefore, the spatial index was not used in the study.

The textual data section is a pop-up section that becomes visible after clicking on a building unit, underground shelter or a land parcel. In the textual data section, which is in the form of a table, it is possible to view all textual and numeric data linked to an object stored within a MongoDB database. For different object types (building unit, underground shelter or land parcel) it shows different fields defined by the data model.

The implemented case study shows that storing and visualization of 3D cadastral data can be performed using the MongoDB database and Cesium JavaScript library. This approach provides easy implementation of the data model and enables basic visualization of stored 3D cadastral data.

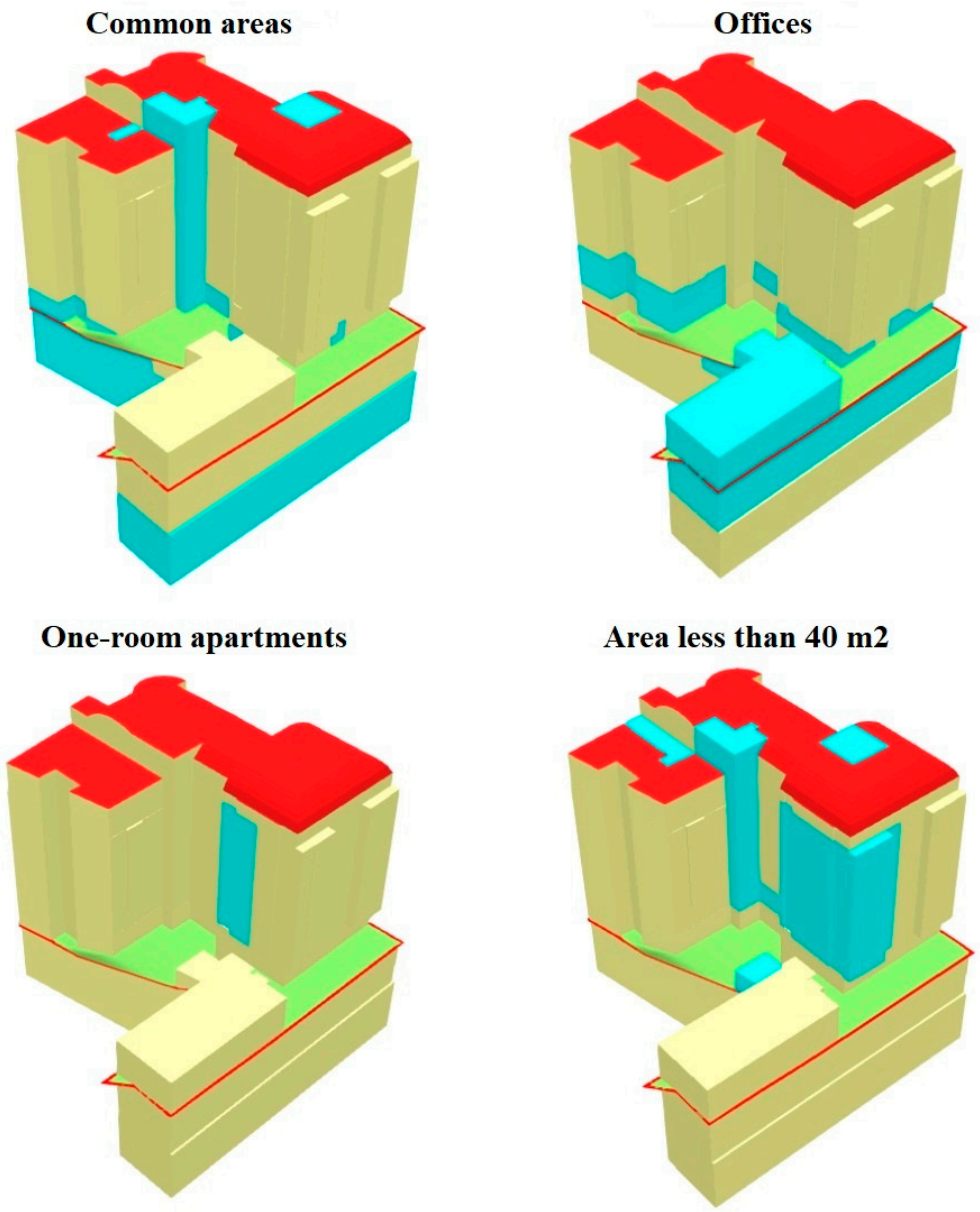

Figure 11. The results of different queries.

4. 3D Cadastre Based on NoSQL and JavaScript Application, Advantages and Shortcomings

The described 3D cadastral prototype, which is based on a NoSQL database and a JavaScript application for 3D visualization, presents a new and innovative use of the developing technologies in the 3D cadastre domain. The prototype and results of the case study indicate that using a NoSQL database for storing 3D cadastre data and JavaScript libraries for 3D visualization and inspection of the data should be considered for the development of a modern 3D cadastral system. However, further development is required in order to fully support 3D cadastre needs. For that reason, it is encouraging and promising that the prototype is based on developing technologies since future technology improvements can be expected. For example, the MongoDB database is a novel database (initial release in 2009) and has been actively developing since then. It is now the leading NoSQL database. Cesium is a JavaScript library that is widely used for 3D visualization, creating 3D globes and 2D maps in a web browser. The library is subject to constant development and new versions are released on a monthly basis. The number of pilots and demonstrators in the land administration domain that use NoSQL databases could be a signal that the significance of this technology will increase

in this domain [38]. At the current stage, the developed prototype of the 3D cadastral system can be described by the following set of advantages and shortcomings.

The main advantage of the prototype based on a NoSQL database is the fact that data are maintained by a Database Management System (DBMS). This provides security, data consistency and data integrity guaranteed by the BASE principles: (1) Basically Available, (2) Soft State, and (3) Eventual consistency [39]. Maintaining cadastral data in other ways, such as a file-based system, cannot guarantee data consistency and data integrity. Furthermore, as it was presented in the case study and the methodology section, the LADM based 3D cadastral data model is easy to implement in the MongoDB database. JSON documents are suitable for storing and exchanging cadastral data defined by the 3D cadastre data model. The implementation of the data model classes as MongoDB collections (sets of JSON documents) is simple and it can be done automatically. Since JSON data are available, additional functionalities on the server and client side can be implemented by using JavaScript language. This includes processing, updating and querying both spatial and textual data. NoSQL databases are designed to support big data and real-time applications, so it can be assumed that they should be able to support a 3D cadastral system containing large data sets of 3D data.

The advantages of the implemented prototype also include the 3D visualization of stored cadastral data. The visualization is performed with an interactive approach. It means that a user can view 3D objects from all sides and can get additional textual and numeric information by clicking on a 3D object. Three different visualization modes are implemented to enable better and intuitive visualization that include: (1) a building unit mode, (2) a floor mode, and (3) a building mode. This is important from the context of visualization requirements. Unlike quite an easy visualization of 2D entities on cadastral maps using 2D geometry, symbols and labels, in 3D cadastre visualization is more complex. Just like in the real world, 3D cadastral objects are overlapping and interlocking. Therefore, a 3D cadastre solution needs to provide better and intuitive visualization of 3D objects. Within the implemented prototype of the system, this is solved through the mentioned three modes. In this way, the system meets the basic requirements of 3D visualization to view and select each 3D object. Beside 3D visualization, the selection of 3D objects based on attribute values is facilitated. In that way, a user is able to query 3D cadastral data and to make an analysis based on these data.

The shortcoming of the 3D cadastral prototype based on a NoSQL database is that BASE principles provide less strict assurance than ACID rules (Atomicity, Consistency, Isolation, Durability), which are followed by relational databases. This is a consequence of NoSQL databases development. To provide additional benefits such as scale, some of the requirements for immediate consistency are relaxed. When it comes to eventual consistency that is usually supported by NoSQL databases, there is a lack of simultaneous updates and usually it takes some time that the system eventually become consistent. That makes NoSQL database model inferior related to the ACID model, but still it could be applied as a less strict consistency model for distributed 3D Cadastre system. However, this shortcoming should be taken into account when implementing a 3D cadastre system based on a NoSQL database.

MongoDB versions prior to version 4.0. offered strong consistency guarantees (strong eventual consistency model). The consistency guarantee is provided only for single document transactions, not for the transactions that contain the updates across multiple documents. The lack of support for multi-document ACID transactions in MongoDB forced the developers to write application-layer code to handle the distributed transactions in MongoDB in cases where this support was required.

However, MongoDB version 4.0. from 2018 provides support for multi-document ACID transaction guarantees [23]. This eliminates the need for developers to write code on application-layer. In accordance with this improvement, it can be concluded that MongoDB can be safely used for the development and implementation of complex distributed systems such as 3D cadastre providing full data consistency guarantees. This kind of development might be expected for other NoSQL databases, so data consistency should not be such an issue.

The lack of topology rules that would provide validation of 3D cadastral data is also the clear shortcoming of the described prototype of the 3D cadastral system. Since the data model is not defined

as a topology model, it is necessary to use topology rules to validate topological consistency. It means that an implemented prototype of the 3D cadastral system currently does not support validation such as checking whether building units intersect or the gaps between building units exist. Since the NoSQL databases (like MongoDB in developed prototype) provide less amount of available spatial functions than relational databases (such as PostgreSQL/PostGIS), the additional development on the server side or using external software and libraries are required to overcome this shortcoming. Furthermore, the system requires additional development on the visualization aspect. Three developed visualization modes provide basic support. However, additional visualization approaches should be supported for full implementation. These approaches should include new context symbology, color selection, transparency, labels, etc. The visualization aspect is very important since it provides a better understanding of stored data and it should be intuitive for both technical and non-technical users.

To provide a more detailed insight in the advantages and shortcomings of the proposed approach, large 3D cadastral datasets should be used for future research. These datasets should include the surrounding 3D cadastral data providing a better context for property visualization. More importantly, the efficiency of the proposed approach for handling large 3D cadastral datasets will be assessed.

5. Conclusions

This study was based on the assumption that an approach of using a NoSQL database and JavaScript application for 3D visualization can be used for the development of a modern 3D cadastral system. A 3D cadastre prototype was developed to achieve the research aim. It was found that a NoSQL database can be used for storing 3D cadastral data defined by a data model based on LADM. In this regard, stored 3D cadastral data are protected by BASE principles. When it comes to 3D visualization, the developed JavaScript application can easily meet basic 3D cadastral requirements such as to view and select each 3D object. The prototype is based on developing technologies and therefore further improvements should be expected.

The prototype itself requires additional development. It is necessary to implement topology validation, since it is very important for cadastral data. Implementing additional, more intuitive visualization approaches is also required. This will enable simpler use of the system for both professionals and non-professionals. In the current stage, it provides maintenance of 3D cadastral data by a DBMS and 3D visualization with the interactive approach and basic support that enables viewing all 3D objects in different modes.

With additional improvements, the prototype can be a good starting point for development of a modern cadastral system that enables 3D registration of real properties, facilitates unequivocal registration of complex 3D property situations, and provides an intuitive user interface and methods for 3D spatial data visualization.

Author Contributions: All authors worked equally on the research conceptualization, investigation, discussion and data modeling. Nenad Višnjevac was involved in data preparation and case study implementation. Rajica Mihajlović, Mladen Šoškić, Željko Cvijetinović, and Branislav Bajat participated in the writing of the manuscript, however Nenad Višnjevac took the lead.

Funding: This research received no external funding.

Acknowledgments: This study was supported by the Serbian Ministry of Education, Science and Technological Development, project No. III 47014. The authors thank the Urbim d.o.o. for providing data used for creating 3D geometry in the case study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Van Oosterom, P. Research and development in 3D cadastres. *Comput. Environ. Urban Syst.* **2013**, *40*, 1–6. [CrossRef]
2. Van Oosterom, P.; Dimopoulou, E. Introduction to the Special Issue: Research and Development Progress in 3D Cadastral Systems. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 59. [CrossRef]

3. Stoter, J.; Salzmann, M. Towards a 3D cadastre: Where do cadastral needs and technical possibilities meet? *Comput. Environ. Urban Syst.* **2003**, *27*, 395–410. [CrossRef]
4. Aien, A.; Kalantari, M.; Rajabifard, A.; Williamson, I.; Wallace, J. Towards integration of 3D legal and physical objects in cadastral data models. *Land Use Policy* **2013**, *35*, 140–154. [CrossRef]
5. Zhang, J.Y.; Yin, P.C.; Li, G.; Gu, H.H.; Zhao, H.; Fu, J.C. 3D Cadastral Data Model Based on Conformal Geometry Algebra. *ISPRS Int. J. Geo-Inf.* **2016**, *5*, 20. [CrossRef]
6. Thompson, R.J.; Van Oosterom, P.; Soon, K.H. LandXML Encoding of Mixed 2D and 3D Survey Plans with Multi-Level Topology. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 171. [CrossRef]
7. ISO TC211. ISO 19152:2012 Geographic Information—Land Administration Domain Model (LADM). Available online: https://www.iso.org/standard/51206.html (accessed on 24 January 2019).
8. Van Oosterom, P.; Lemmen, C. The Land Administration Domain Model (LADM): Motivation, standardisation, application and further development. *Land Use Policy* **2015**, *49*, 527–534. [CrossRef]
9. Lemmen, C.; Van Oosterom, P.; Bennett, R. The Land Administration Domain Model. *Land Use Policy* **2015**, *49*, 535–545. [CrossRef]
10. Navratil, G. Cadastral Boundaries: Benefits of Complexity. *J. Urban Reg. Inf. Syst. Assoc.* **2011**, *23*, 19–27.
11. Stoter, J.; Van Oosterom, P. Incorporating 3D geo-objects into a 2D geo-DBMS. In Proceedings of the ASPRS/ACSM Conference, Washington, DC, USA, 19–26 April 2002.
12. Shojaei, D.; Olfat, H.; Faundez, S.I.Q.; Kalantari, M.; Rajabifard, A.; Briffa, M. Geometrical data validation in 3D digital cadastre—A case study for Victoria, Australia. *Land Use Policy* **2017**, *68*, 638–648. [CrossRef]
13. Fu, L.; Yin, P.; Li, G.; Shi, Z.; Liu, Y.; Zhang, J. Characteristics and Classification of Topological Spatial Relations in 3-D Cadasters. *Information* **2018**, *9*, 71. [CrossRef]
14. Breunig, M.; Zlatanova, S. 3D geo-database research: Retrospective and future directions. *Comput. Geosci.* **2011**, *37*, 791–803. [CrossRef]
15. Janečka, K.; Karki, S.; van Oosterom, P.; Zlatanova, S.; Kalantari, M.; Ghawana, T. 3D Cadastres Best Practices, Chapter 4: 3D Spatial DBMS for 3D Cadastres. In Proceedings of the FIG Congress 2018, Istanbul, Turkey, 6–11 May 2018.
16. Shojaei, D.; Kalantari, M.; Bishop, I.D.; Rajabifard, A.; Aien, A. Visualization requirements for 3D cadastral systems. *Comput. Environ. Urban Syst.* **2013**, *41*, 39–54. [CrossRef]
17. Shojaei, D.; Olfat, H.; Rajabifard, A.; Briffa, M. Design and Development of a 3D Digital Cadastre Visualization Prototype. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 384. [CrossRef]
18. Navratil, G.; Schwai, M.; Vollnhofer, S.; Konturek, P.; Giannopoulos, I. From Floor Plans to Condominium Rights Through an Augmented Reality Approach. In Proceedings of the 6th International FIG 3D Cadastre Workshop, Delft, The Netherlands, 2–4 October 2018.
19. Belussi, A.; Migliorini, S.; Eldawy, A. Detecting Skewness of Big Spatial Data in SpatialHadoop. In Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Seattle, WA, USA, 6–9 November 2018.
20. Belussi, A.; Migliorini, S.; Negri, M.; Pelagatti, G. Validation of spatial integrity constraints in city models. In Proceedings of the Fourth ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems (MobiGIS 15), Bellevue, WA, USA, 3–6 November 2015.
21. R Language and Environment for Statistical Computing and Graphics. Available online: https://www.r-project.org/about.html (accessed on 22 April 2019).
22. Cesium JavaScript Library. Available online: https://cesiumjs.org/index.html (accessed on 31 January 2019).
23. MongoDB. Available online: https://www.mongodb.com/ (accessed on 25 April 2019).
24. NoSQL Viewer. Available online: http://www.spviewer.com/nosqlviewer.html (accessed on 22 April 2019).
25. QGIS. Available online: https://qgis.org/en/site/ (accessed on 22 April 2019).
26. Višnjevac, N.; Mihajlović, R.; Šoškić, M.; Cvijetinović, Ž.; Marošan, S.; Bajat, B. Developing Serbian 3D Cadastre System—Challenges and Directions. In Proceedings of the 6th International FIG 3D Cadastre Workshop, Delft, The Netherlands, 2–4 October 2018.
27. Lemmen, C.; van Oosterom, P.; Thompson, R.J.; Hespanha, J.; Uitermark, H. The Modelling of Spatial Units (Parcels) in the Land Administration Domain Model (LADM). In Proceedings of the XXIV FIG International Congress, Sydney, Australia, 11–16 April 2010.
28. DB-Engines Ranking. Available online: https://db-engines.com/en/ranking (accessed on 28 January 2019).

29. Višnjevac, N.; Mihajlović, R.; Šoškić, M.; Cvijetinović, Ž.; Bajat, B. Using NoSQL databases in the 3D cadastre domain. *Geod. Vestn.* **2017**, *61*, 412–426. [CrossRef]
30. Murshed, S.M.; Al-Hyari, A.M.; Wendel, J.; Ansart, L. Design and Implementation of a 4D Web Application for Analytical Visualization of Smart City Applications. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 276. [CrossRef]
31. Chen, Y.; Shooraj, E.; Rajabifard, A.; Sabri, S. From IFC to 3D Tiles: An Integrated Open-Source Solution for Visualising BIMs on Cesium. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 393. [CrossRef]
32. Zhu, L.; Wang, Z.; Li, Z. Representing Time-Dynamic Geospatial Objects on Virtual Globes Using CZML—Part I: Overview and Key Issues. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 97. [CrossRef]
33. Zhu, L.; Li, Z.; Wang, Z. Representing Time-Dynamic Geospatial Objects on Virtual Globes Using CZML—Part II: Impact, Comparison, and Future Developments. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 102. [CrossRef]
34. Cemellini, B.; Thompson, R.; Van Oosterom, P.; De Vries, M. Usability testing of a web-based 3D Cadastral visualization system. In Proceedings of the 6th International FIG 3D Cadastre Workshop, Delft, The Netherlands, 2–4 October 2018.
35. Digital Platform of National Spatial Data Infrastructure—GeoSrbija. Available online: https://a3.geosrbija.rs/ (accessed on 3 February 2019).
36. Digital Real Estate Cadastre Database—EKatastar. Available online: http://katastar.rgz.gov.rs/KnWebPublic/PublicAccess.aspx (accessed on 3 February 2019).
37. Constructive Solid Geometry (CSG) JavaScript library. Available online: https://github.com/evanw/csg.js/ (accessed on 7 February 2019).
38. Bennett, R.M.; Pickering, M.; Sargent, J. Transformations, transitions, or tall tales? A global review of the uptake and impact of NoSQL, blockchain, and big data analytics on the land administration sector. *Land Use Policy* **2019**, *83*, 435–448. [CrossRef]
39. Chandra, D.G. BASE analysis of NoSQL database. *Future Gener. Comput. Syst.* **2015**, *52*, 13–21. [CrossRef]