Universitatea Tehnică de Construcții București
Școala Doctorală

Faculty of Geodesy
Surveying and Cadastre Department
Research Centre:
Geodetic Engineering Measurements and Spatial Data Infrastructures

RESEARCH REPORT II

Study of Spatial Modelling Techniques

PhD. Supervisor

Professor PhD. Eng. Petre Iuliu DRAGOMIR

PhD. Candidate

Eng. Anca Patricia GRĂDINARU

Bucharest
2022

Table of Contents

1. Introduction

Models that represent the geometric elements of a city, in a three-dimensional space, are called 3D models of the city and can faithfully describe the environment. Other types of data, such as statistical, social, historical data, etc., can be associated with geometric data, leading to a holistic approach to the spatial representation of cities, as well as to the sustainable development and management of cities. Thus, 3D semantic city models are created, which include, besides geometric information, semantic information or knowledge about the represented urban area (Billen et al., 2014).

The need to develop 3D spatial models of cities is constantly increasing, these models being useful in a wide variety of fields. They allow, in addition to visualization, to carry out a series of analyses and obtain statistical data. In a study conducted by Biljecki et al. (2015), the usefulness of 3D models of cities in a large number of areas is highlighted, such as solar radiation estimation, energy demand estimation, positioning, building classification, improvement of visualization methods, visibility analysis, estimation of shadow areas produced by buildings, noise propagation estimation, 3D cadastre, urban planning, emergency response, population estimation, disaster prediction, archaeology, etc. (*Figure 1.1*). The data necessary for such representations are acquired through different methods, such as laser scanning, photogrammetry, extrusion of buildings footprints, but also architectural models and drawings (Biljecki et al., 2015).

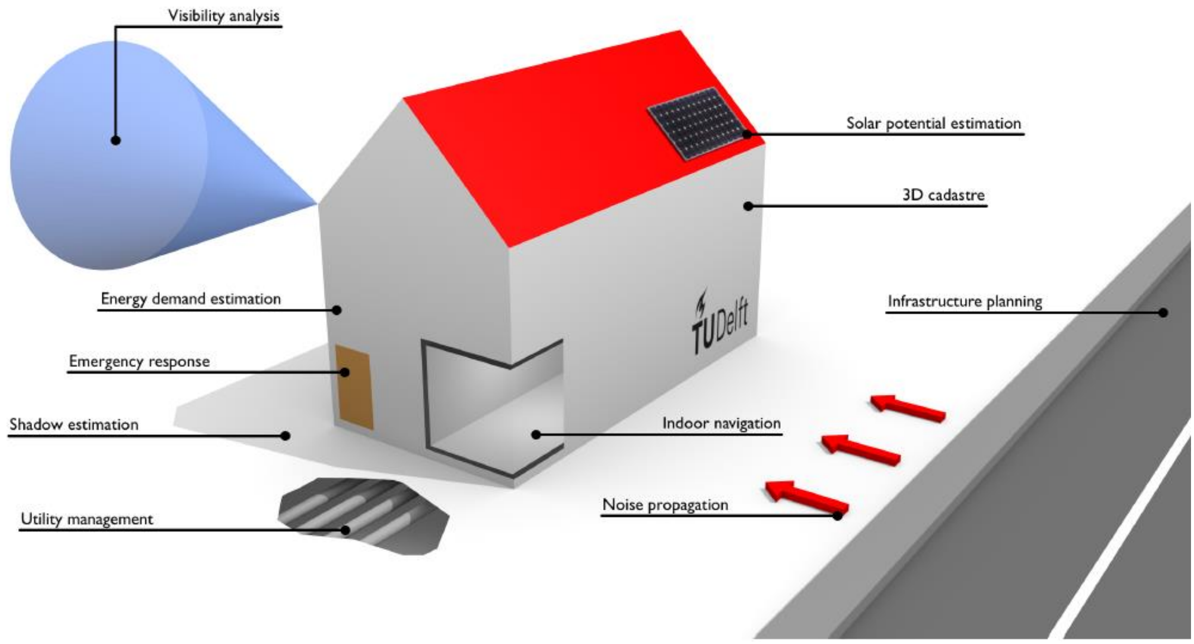


Figure 1.1. Examples of Applications of 3D City Models (Biljecki et al., 2015)

In terms of spatial modelling techniques, traditionally, generating 3D models of cities required a large amount of work, consisting of scanning maps and acquiring digital images, obtaining buildings footprints and manually extruding them, depending on the height of each construction, or manually modelling 3D geometries using specialized software products.

Currently, to reduce the workload and time invested, software products are used to automatically generate the 3D models of the cities (Takase et al., 2003).

An example of an automatically generated 3D spatial representation of buildings can be the 3D BAG (*Figure 1.2*). This is an open data set containing 3D models of buildings in the Netherlands, with three different levels of detail and constantly updated. The data leading to this model is obtained from the Dutch Building and Address Register (BAG), which is a set of data relating to the address of each building, as well as its destination and date of construction. This dataset is updated as buildings are constructed or demolished. Data are also taken from the Dutch National Height Model (AHN), obtained by laser scanning, with an average density of 8 points per square meter (Overview - 3D BAG).

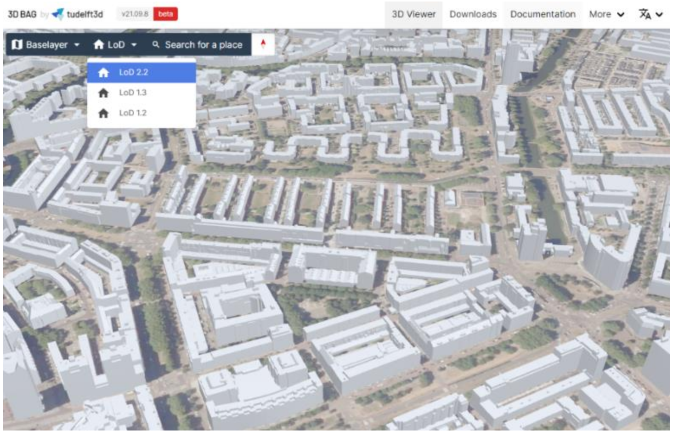


Figure 1.2. 3D Representation of Buildings in Rotterdam, Netherlands, from the 3D Bag Dataset (Overview - 3D BAG)

Virtual Singapore (*Figure 1.3*) is a three-dimensional semantic and dynamic model of cities in Singapore that includes detailed information such as the texture of represented objects, land attributes, vegetation, water, transportation, infrastructure, etc. It is built based on 2D geometric data and data from various public agencies (Virtual Singapore).

Figure 1.3. Virtual Singapore - Building a 3D-Empowered Smart Nation (GWPrime)

Chosing the modelling technique is influenced by the level of detailed required for the purpose of the project, and also by the method of spatial data acquisition, such as laser scanning or photogrammetric methods. In this report, different 3D spatial modeling techniques will be studied, using data from different sources, such as point clouds or orthorectified images, but also different software products that allow the creation and visualization of these models.

2. Software Products and 3D Spatial Modelling Methods

2.1. 3D Model Generation Methods Based on Point Clouds

According to Maas & Vosselman (1999), the main methods of automatically generating 3D models of buildings from point clouds obtained by laser scanning are: model-driven methods for generating 3D models and data-driven methods for generating 3D models.

In the case of model-driven methods for generating 3D models, we use primitives which are simple shapes, described by a set of parameters, found in a library of models. Thus, for simple buildings, the library is searched for the model closest to the building and the most likely values of the parameters are calculated. For complex buildings, one method is to divide the complex form of the building into several primitives, which can be done based on the roof break lines (Tarsha-Kurdi et al., 2007).

Methods for generating data-driven 3D models assume that any building is a polyhedron and involve modelling without dividing the construction into several primitives, analysing the point cloud entirely (Tarsha-Kurdi et al., 2007).

In a comparative study of the two methods, Tarsha-Kurdi et al. (2007) concluded that the main advantage of the model-based method is that it allows geometric models to be obtained quickly without visual distortions, while data-driven methods lead to polyhedral 3D models, which more faithfully represent the reality. The disadvantage of the latter is that some visual deformations occur, which can be reduced by a homogeneous distribution of the point cloud and a suitable density of points, closely related to the size of the elements to be modelled.

2.2. CityGML Standards

2.2.1. General Characteristics

CityGML standards define a semantic conceptual model and a means of data exchange for representing and storing 3D spatial models of the cities, contributing to the integration of the geospatial data from urban areas into a large number of applications. CityGML is based on GML (Geography Markup Language), which is an XML format used to model geometric features (CityGML | OGC).

CityGML has been developed since 2002 as an open-data model and defines both urban object classes and their relationships. Classes of objects can define not only buildings, but also vegetation, elevation, waters, as well as "urban furniture" (Gröger et al., 2008). Thus, 3D models of cities are represented both geometrically and topologically, as well as through databases containing relevant information about the represented urban areas.

The CityGML model consists of a main module and a number of thematic extensions. The main module comprises the essential components of the CityGML data model, and based on it each extension defines certain thematic 3D spatial models, which can be used in different combinations alongside the main module, forming CityGML profiles. The main thematic extensions implemented by the CityGML standard are: *Appearance*, *Building*, *CityFurniture*, *CityObjectGroup*, *Generics*, *LandUse*, *Relief*, *Transportation*, *Vegetation*, *WaterBody* and *TexturedSurface* (Gröger et al., 2008).

Currently, CityGML 3.0 is implemented, which is an evolution of the previous two versions (1.0 and 2.0) and allows data encoding both in GML/XML format and in JSON format (JavaScript Object Notation) or database schema.

2.2.2. Level of Detail

The term "Level of Detail" (LOD) is used to define how complete the data in the composition of the 3D model are (Biljecki et al., 2014), or the degree of proximity of the model to the reality (Tang et al., 2020). CityGML 2.0 standards define 5 levels of detail (*Figure 2.1*) as it follows (Gröger et al., 2006):

- LOD0 is a 2.5D digital model of the terrain, over which a photogrammetric image or map can be draped;
- LOD1 is a model that includes buildings represented by rectangular shapes, with flat roofs;
- LOD2 additionally includes differentiated roof shapes as well as thematic surfaces and vegetation elements;
- LOD3 represents architectural models with detailed elements of walls, roofs and balconies;
- LOD4 complements the LOD3 with the interior structures of the buildings.

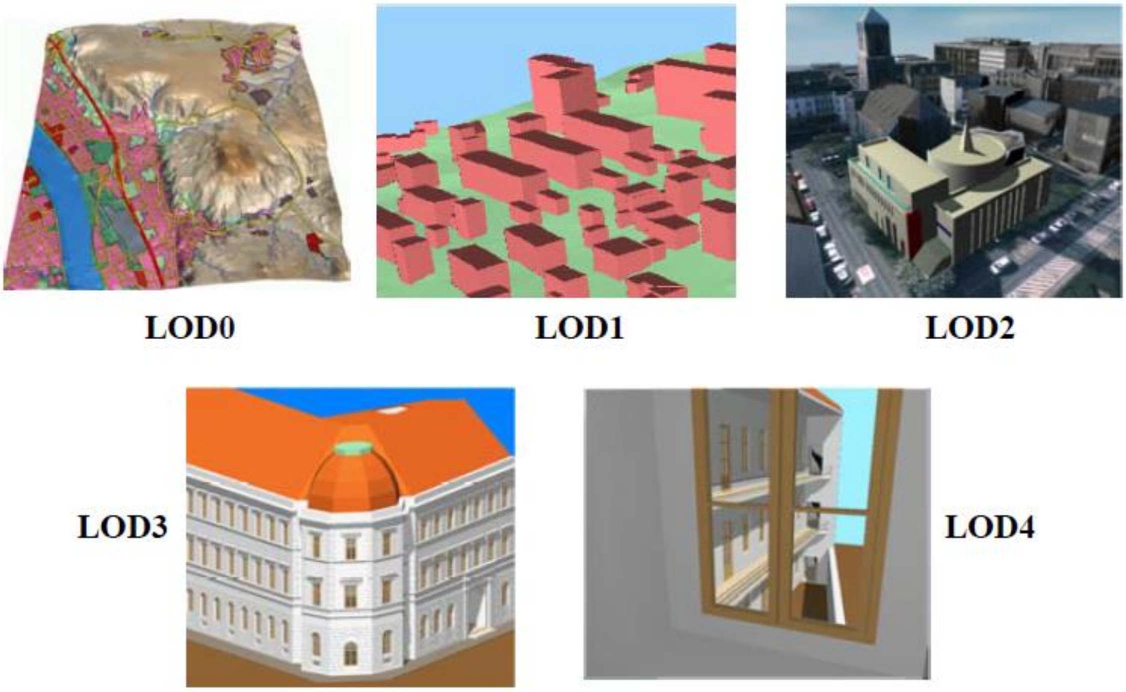


Figure 2.1. Levels of Detail in CityGML 2.0 (Kolbe et al., 2005)

Biljecki et al. (2016) concluded that the five levels of detail are insufficient from a geometric perspective and redefined the concept of LOD as shown in *Figure 2.2*.

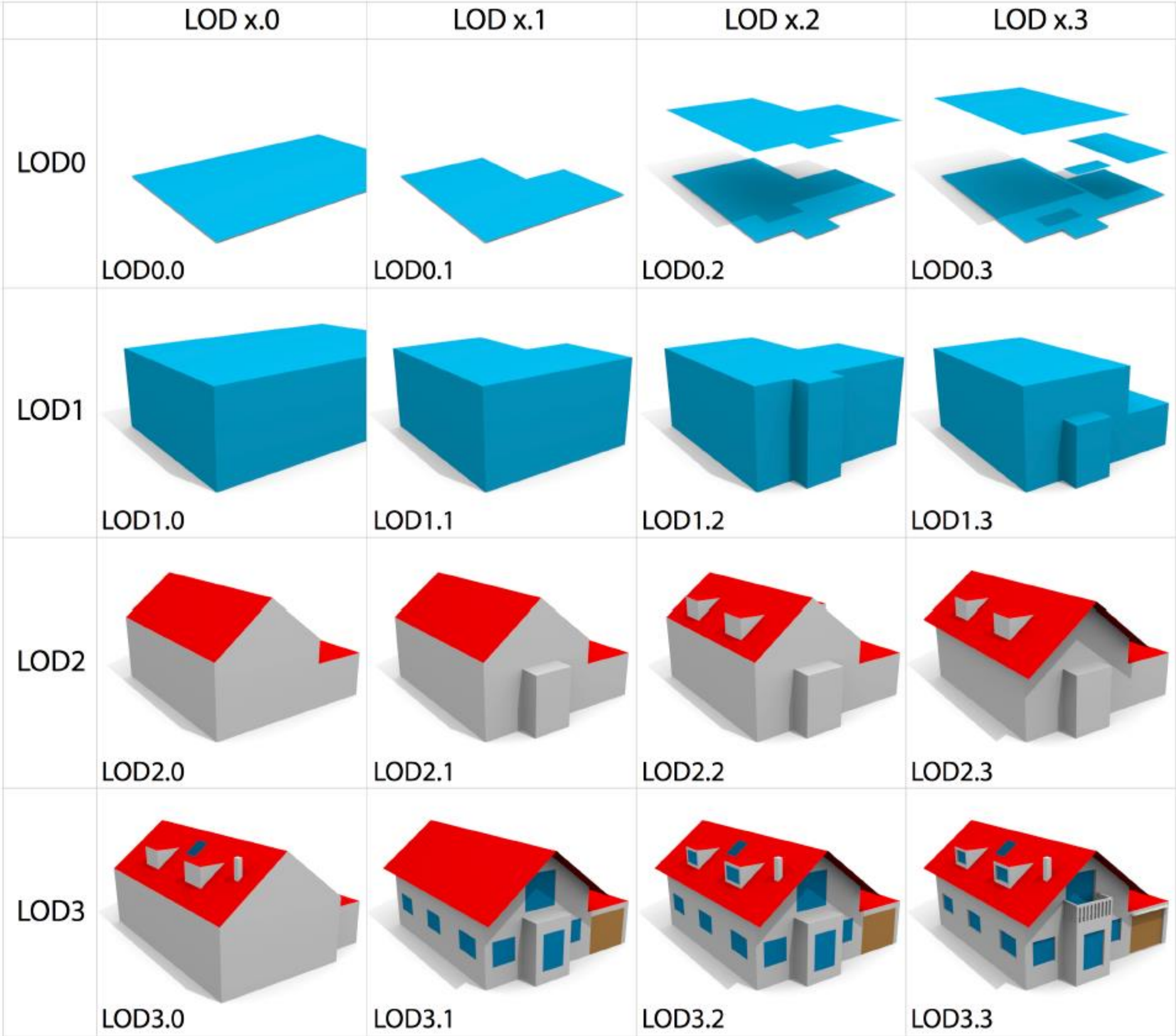


Figure 2.2. The 16 Levels of Detail Defined by Biljecki et al. (2016)

According to Biljecki et al. (2016), different spatial data acquisition techniques lead to different spatial representations of the objects and implicitly to a different level of detail of the obtained models, even though according to CityGML 2.0 standards, they fall into the same category. Their study focuses on the exterior of buildings and highlights the importance of the concept of LOD in numerous applications, in addition to the initial visualization one, hence the importance of fragmenting the existing 5 levels of detail.

In the case of CityGML 3.0, the concept of LOD has been modified so that each level of detail includes a representation of interior objects, and LOD4 has been excluded (Kutzner et al., 2020).

2.2.3. Modelling Principles

A basic principle in modeling according to CityGML standards is the coherent connection between geometric/topological elements and their attributes (*Figure 2.3*). Semantically, features are represented by different attributes and relationships, and spatially, objects are represented by 3D geometries. Thus, the model is divided into two hierarchical structures: the semantic model and the geometric model (Stadler & Kolbe, 2012).

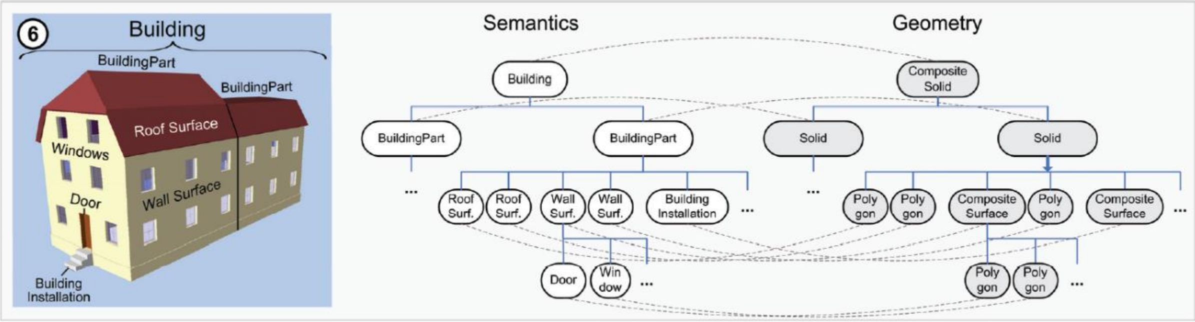


Figure 2.3. Complex Construction with Spatio-Semantic Structure (Stadler & Kolbe, 2012)

Another important aspect of modelling is the integration of 3D objects and the terrain they are on. In some cases, 3D objects are represented above or submerged in the terrain, and a *TerrainIntersectionCurve* (*Figure 2.4*) is used to solve this problem, suggesting the line of intersection between the terrain and the represented element (Gröger et al., 2008). This is very important in certain analyses, such as calculating solar radiation or estimating noise propagation, where only objects located on the surface of the terrain are relevant (Gröger & Plümer, 2012).

Objects that are not relevant to the intended purpose are not defined by a 3D volume, but are represented using *ClosureSurfaces* (Figure 2.4). These surfaces can also be used to represent objects that are not on the land surface, such as tunnels or underpasses, which are needed to perform analyses such as flood simulations (Gröger et al., 2008).

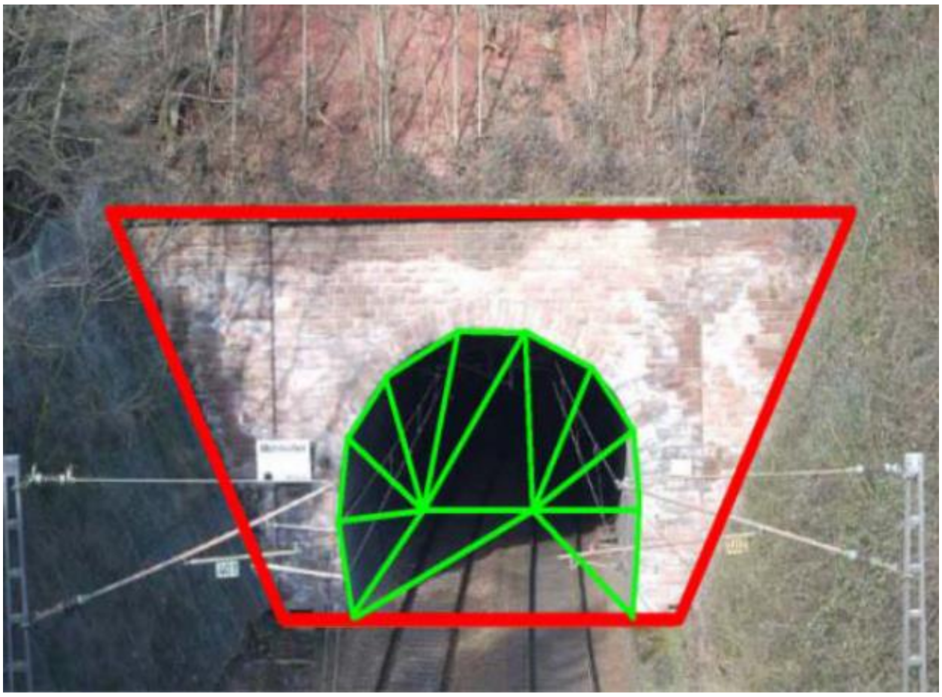


Figure 2.4. Tunnel Modeling with TerrainIntersectionCurve (red) and ClosureSurfaces (green) (Kolbe et al., 2005)

In order to avoid mistakes in the classification of the objects by their attributes and to ensure the interoperability of data, *ExternalCodeLists* are used, which are implemented using dictionaries (in CityGML 3.1.1.: *Simple Dictionary Profile*) in which all types of attributes are included (Gröger et al., 2008).

External references are used for situations where the represented 3D objects come from external datasets, e.g. architectural models, or when there are relationships between them and certain databases, e.g. cadastral databases, and are particularly useful for updating data (Gröger et al., 2008).

In addition to spatial and semantic aspects, modelled 3D objects are also characterized by observable properties of external surfaces (*Appearances*), which represent both surface textures and different themes such as infrared radiation, noise pollution, etc. They contribute to improving visualization as well as data analysis processes (Gröger & Plümer, 2012).

In some situations, CityGML may have certain limitations, which is why the model is extended with the help of an ADE (*Application Domain Extensions*) mechanism. Such a mechanism allows the addition of new object classes, new attributes or new standard geometries for certain object classes (Biljecki et al., 2021).

2.3. ArcGIS CityEngine ESRI Software Product

2.3.1. General Characteristics

CityEngine is an application developed by ESRI used for 3D spatial modelling of urban areas, based on geospatial data imported in an Esri geodatabase (.gdb) or shapefile (.shp) format (Advanced 3D City Design Software | ArcGIS CityEngine).

The basis of modelling with the the CityEngine software product is a procedural approach to the work stages, which leads to an increased efficiency of the modelling process, eliminating manual interaction between the user and the program by entering a code consisting of a series of commands to be executed by the program. The programming language is CGA shape grammar, designed specifically for generating 3D architectural models. The main idea is to define rules that are applied iteratively, generating more details (*Figure 2.5*). Building modelling is accomplished in the following steps (About CityEngine—ArcGIS CityEngine Resources | Documentation):

- Buildings footprints are either generated by CityEngine or imported;
- The user selects the .cga file with the rules to be applied either to all constructions or individually;
- The user can start the modelling process by applying the established rules on the selected geometries;
- To edit the resulting model, you can edit the applied rules, or modify the parameters of individual buildings;
- After completing the modelling process, the result can be exported.

Figure 2.5. The Iterative Process of Defining Buildings Details in CityEngine Software (About CityEngine—ArcGIS CityEngine Resources | Documentation)

2.3.2. Modelling Principles

The input data required in modelling with the CityEngine software product, as evidenced by Turksever (2015) can be: the digital terrain model (DTM), or the digital elevation model (DEM), in TIFF format, road axes, with line geometries and a table of attributes containing street names, types of roads, sidewalks, speed limits, etc., buildings footprints, represented by polygon geometries and their attribute table that can include types of buildings, building heights, number of floors, vegetation and water areas, etc..

In the modelling process, different tools are used that make up the workflow represented in *Figure 2.6*. For the first step, the necessary data is imported and the road network is generated. The areas bounded by roads are then divided to create the lots on which the buildings are located, followed by the generation of the building geometries (Parish &; Müller, 2001).

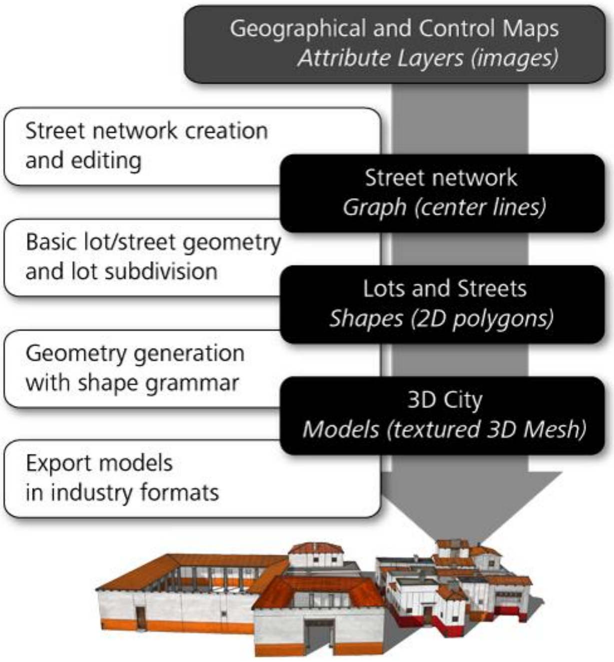


Figure 2.6. The Mechanism for Generating 3D Models with the CityEngine Software Product (About CityEngine—ArcGIS CityEngine Resources | Documentation)

To create the road network, files in *.DXF, *.GDB, *.OSM or *.SHP format can be imported, containing street segments (line) and network nodes (Turksever, 2015). For this, there is the

Grow streets tool that allows, in addition to creating a new street network, expanding existing networks. The algorithm allows differentiation between larger streets, which enclose a block, and smaller streets within them (Generation of street networks—ArcGIS CityEngine Resources | Documentation). There are three types of street network templates, represented in *Figure 2.7*.

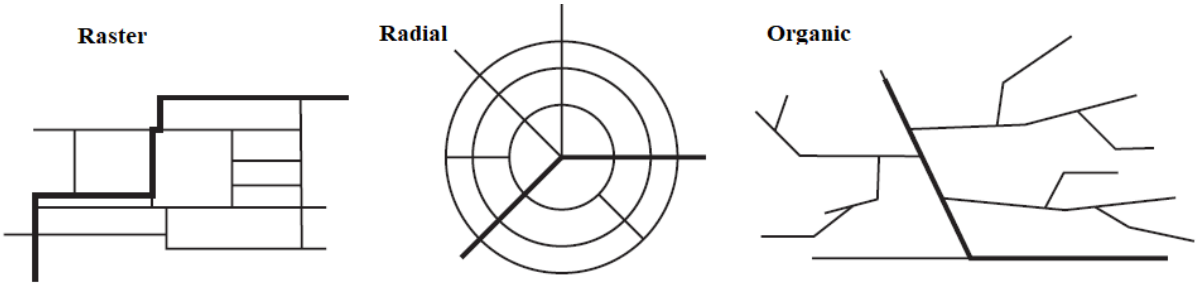


Figure 2.7. Types of Street Networks that Can Be Represented with the CityEngine Software Product (Parish &; Müller, 2001)

To adapt the street network to the digital elevation model (*Figure 2.8*), the terrain must be selected and the adaptation function enabled (Generation of street networks—ArcGIS CityEngine Resources | Documentation).

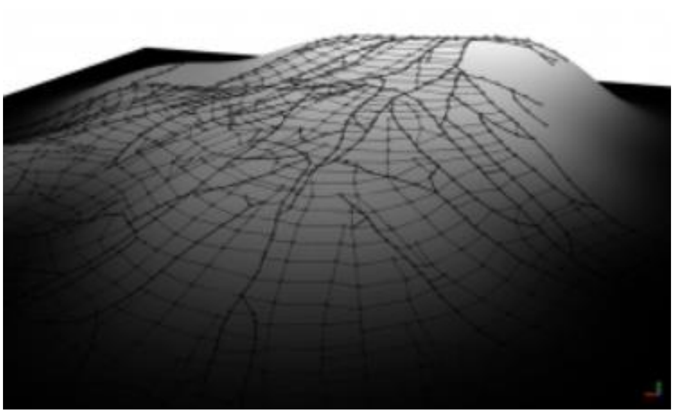


Figure 2.8. Adapting the Street Network to Land Elevation in CityEngine (Generation of street networks—ArcGIS CityEngine Resources | Documentation)

Other tools that can be used are *Edit street/curves* (for editing the width of the street and sidewalks, as well as their curvature), *Clean-up streets* (for joining street segments and creating nodes at their intersection), *Align streets to terrain* (for aligning the network with map layers that have the defined elevation), *Generate Bridges* (useful for models that do not have elevation data and it is necessary to represent different passages in 3D), *Fit widths to shapes* (for datasets that do not contain attributes that define street widths) or *Simplify graph* (for transforming polylines with many inflection points into longer, curved lines).

For building modelling, their footprints can be imported in shapefile format, accompanied by attributes that define the height of the buildings (Turksever, 2015). To model them, the file in .cga format is necessary, which contains the rules to be applied. The process is iterative, which means that at each stage a geometry is taken and modified to obtain a new geometry. In creating the .cga file, conditions can be placed related to the heights of the buildings (*attr minheight*=..., *attr maxheight*=...) or to the extrusion of buildings footprints according to the established height (*lot --> extrude*) (Work with rules—ArcGIS CityEngine Resources | Documentation).

Operations that can be performed with imported geometries are (*Figure 2.9*) *Extrusion* (for the extrusion of a geometry at a given height), *Transformation* (places, ranks and sizes the geometry), *Component split* (divides a geometry into several components), *Subdivision split* (divides geometry along axes), *Insert assets* (loads different elements and inserts them into the created model) (Essential shape operations overview—ArcGIS CityEngine Resources | Documentation).

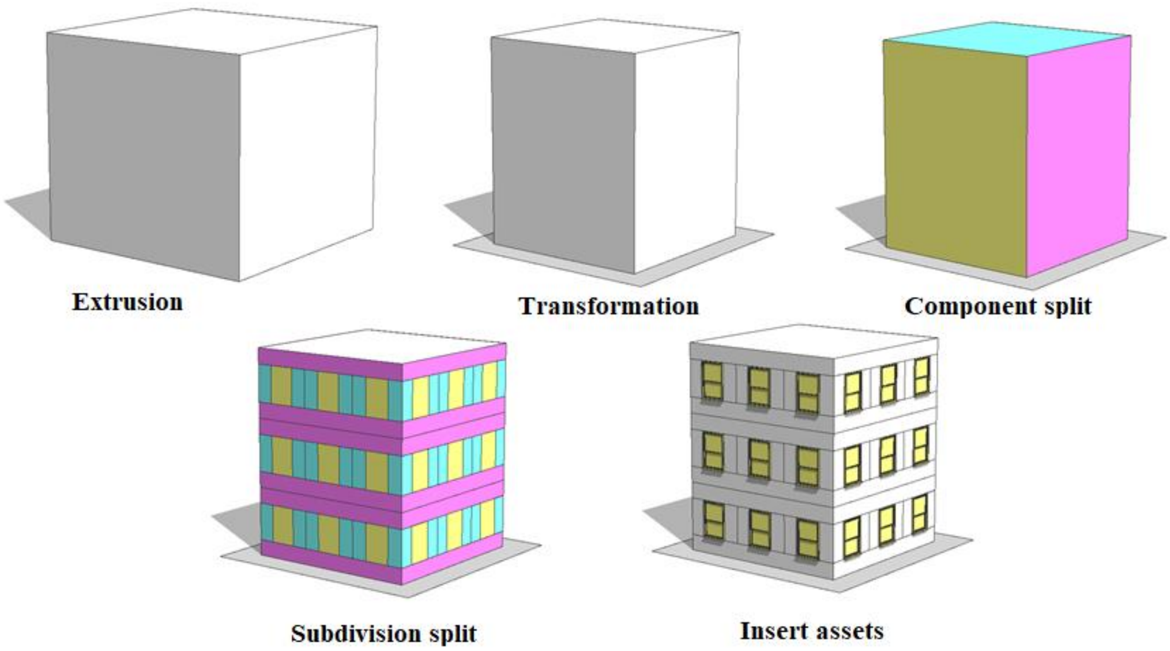


Figure 2.9. Essential Shape Operations Overview—ArcGIS CityEngine Resources | Documentation)

To apply textures to building facades and roofs, different façade images or rooftop satellite images can be used, which are also found in Esri libraries.

The results obtained from the modelling process can be exported to a CityEngine WebScene format, viewed and distributed online via CityEngine web viewer, in .3ws format (*Figure 2.10*).

Figure 2.10. 3D Model Viewed in CityEngine Web Viewer (CityEngine Web Viewer - Kisumu3D)

2.4. Esri 3DCIM – 3D City Information Model Solution

2.4.1. Introduction

The 3D City Information Model (3DCIM) is a model developed by Esri for data management, analysis, and visualization (*Figure 2.11*). 3D models of cities can be brought into ArcGIS and CityEngine and can be used in a wide range of applications. 3DCIM and CityGML are complementary and interoperable (Maren, 2014).

Figure 2.11. Esri 3DCIM Solution (Maren, 2014)

2.4.2. Modelling Principles

The 3DCIM model is compatible with CityGML standards and with the data format required to ensure interoperability of models. From a content perspective, the 3DCIM model includes three main themes (Reitz & Schubiger-Banz, 2014):

- **Built environment** (*Figure 2.12*) – consisting of features and networks such as structures (buildings, bridges, tunnels), interior elements of buildings, installations (hydrants, poles, rubbish bins, bicycle racks), utilities, transport networks, etc.; the *Building* element represents the footprint of the building and contains the most attributes, *BuildingShell* is used for the 3D representation of the building body, and *BuildingShellParts* is used in the process of separating different elements such as roofs or walls (Reitz et al., 2014);

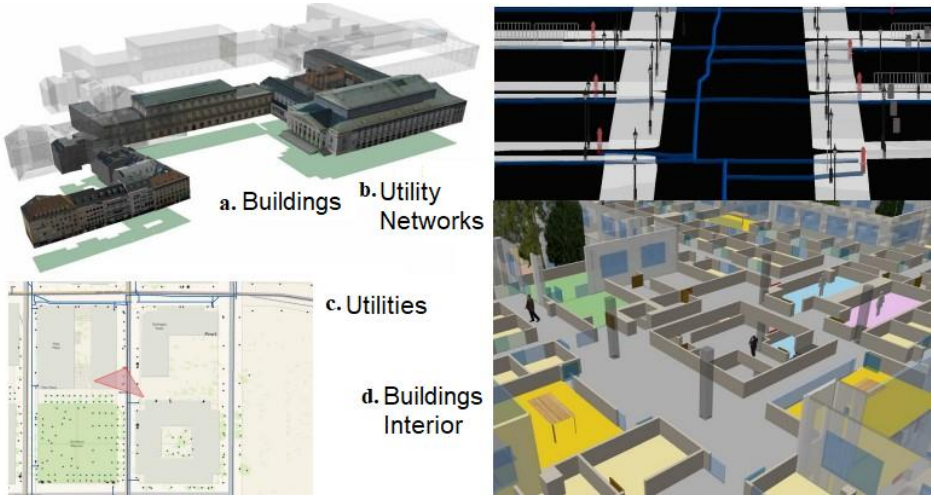


Figure 2.12. The Built Environment in 3DCIM (Reitz et al., 2014)

- **Legal environment** (*Figure 2.13*) – includes elements of urban zoning, regulations (e.g. maximum allowed height of buildings) and property boundaries;

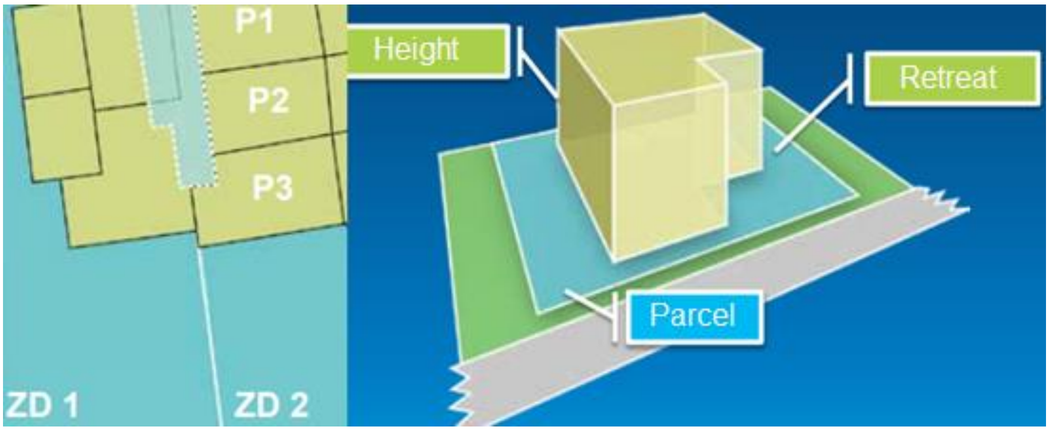


Figure 2.13. Legal Environment in 3DCIM (Reitz et al., 2014)

To this end, there's also the ArcGIS Urban application, which enables digital city transformation and regional planning (Urban Planning and Design—Smart Urban Planning | ArcGIS Urban).

- **Natural environment** (*Figure 2.14*) – includes natural features such as land coverage with vegetation or water, but also geological structures, atmosphere, weather or climate.

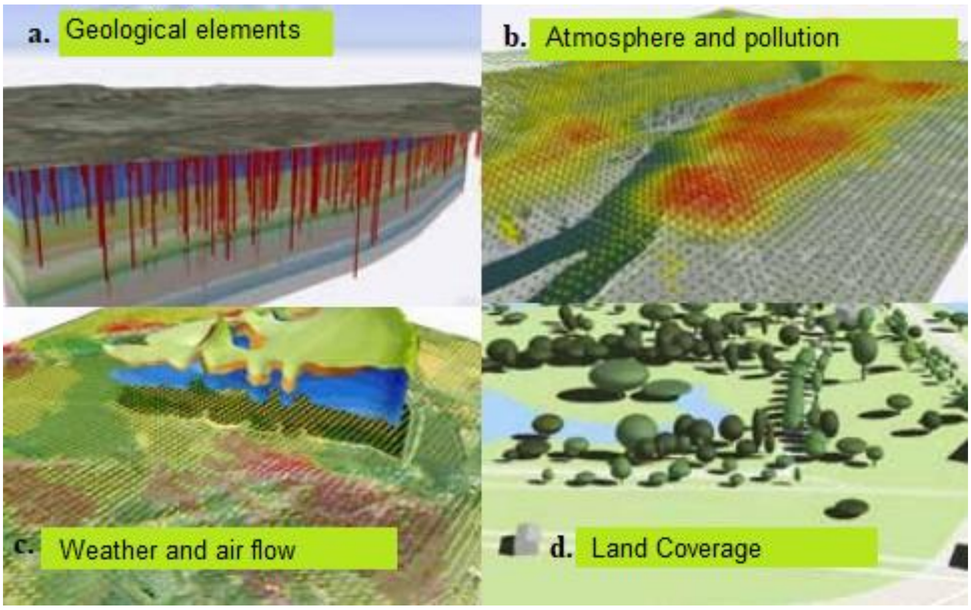


Figure 2.14. The Natural Environment in 3DCIM (Reitz et al., 2014)

2.5. 3D Spatial Modelling *3dfier* Software Product

The *3dfier* software product is an open-source 3D spatial modelling solution, developed by a group of researchers from the Delft University of Technology, based on an automated process for reconstructing 3D objects. It uses 2D geographic data, represented by polygons, and based on a point cloud, the elevation and semantic elements necessary for the extrusion of 2D geometries are obtained (*Figure 2.15*). The ultimate goal is to achieve error-free, overlap-free and gap-free surfaces (Ledoux et al., 2021).

The input data used in modelling are 2D polygons representing buildings, lakes, roads, etc. and elevation points, usually obtained from laser scanning, in *.LAS format, or extracted from photogrammetric images. Each polygon is assigned a specific class: *Terrain*, *Forest*, *Water*, *Road*, *Building*, *Bridge/Overpass* and *Separation* (for walls or fences). The semantic data corresponding to each polygon is used to obtain a 3D geometry by extrusion. The obtained geometries are joined to form a single surface, without voids and overlaps (Ledoux et al., 2021).

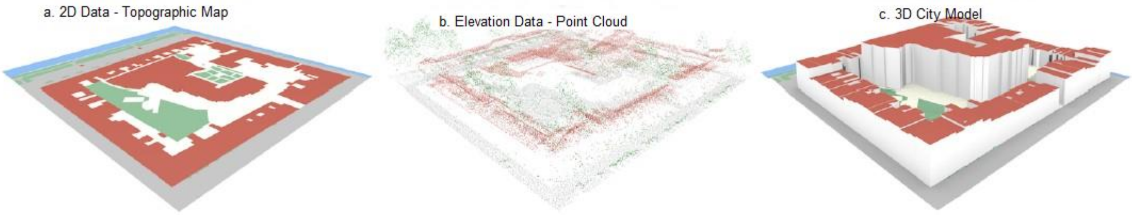


Figure 2.15. 3D Spatial Modelling with the 3dfier Software Product (Ledoux et al., 2021)

2.6. Using ArcGIS Pro Software Product in 3D Spatial Modelling

2.6.1. Introduction

ArcGIS Pro is a GIS application that allows integration, visualization, edit, analysis and distribution of geospatial data in 2D, 3D or 4D spaces, being compatible with a number of other ArcGIS products, such as ArcGIS Online or ArcGIS Enterprise (2D, 3D & 4D GIS Mapping Software | ArcGIS Pro).

2.6.2. Modelling Principles

Within ArcGIS Pro, data is stored in a geospatial database, which can include vector, raster, or attribute features (*Figure 2.16*). Vector data is stored on layers called *Feature Classes*. They are a collection of features that share the same geometry (*point, line*, or *polygon*), attributes, and coordinate reference system. A *Feature Dataset* is a collection of classes of elements that are stored together and have the same spatial reference. A class of items that stores textual data or representations that contain information about objects is called *Annotation* (ArcGIS Pro Resources | Tutorials, Documentation, Videos & More).

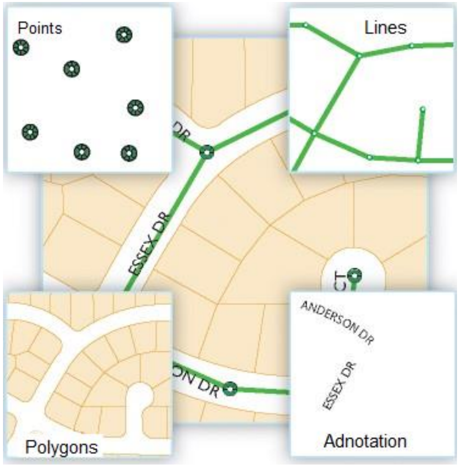


Figure 2.16. Object Types in GIS (ArcGIS Pro Resources | Tutorials, Documentation, Videos & More)

Raster data (*Figure 2.17*) is stored in raster datasets, with each image stored on a layer. Raster datasets can be stored in many formats, such as *.TIFF, MrSid, ESRI Grid, etc.. Raster data can serve as a base-map for the area to be modelled (ArcGIS Pro Resources | Tutorials, Documentation, Videos & More).

Figure 2.17. Raster Image in ArcGIS Pro (ArcGIS Pro Resources | Tutorials, Documentation, Videos & More)

Within ArcGIS Pro, by using the *Local Scene* template, a 3D scene of the project, in a projected coordinate system, where the curvature of the Earth is not taken into consideration, can be created. A local scene with 3D objects can be used to represent and visualize objects such as buildings or 3D models of cities. For 3D building editing (*Figure 2.18*), the building footprint can be converted into a multipatch feature. Data from LiDAR point clouds can be used to obtain the correct buildings heights. A point cloud can be imported into ArcGIS Pro in either ASCII or LAS format to represent different constructions or details of them. Within the representation, symbology can be modified and textures can be added to elements for a more realistic representation. Textures can be added from the *Edit* tab, with the *Multipatch* texture function. For the representation of the buildings, textures from the real world, for the facades of the buildings and for the roof can be added (Badea, 2020).

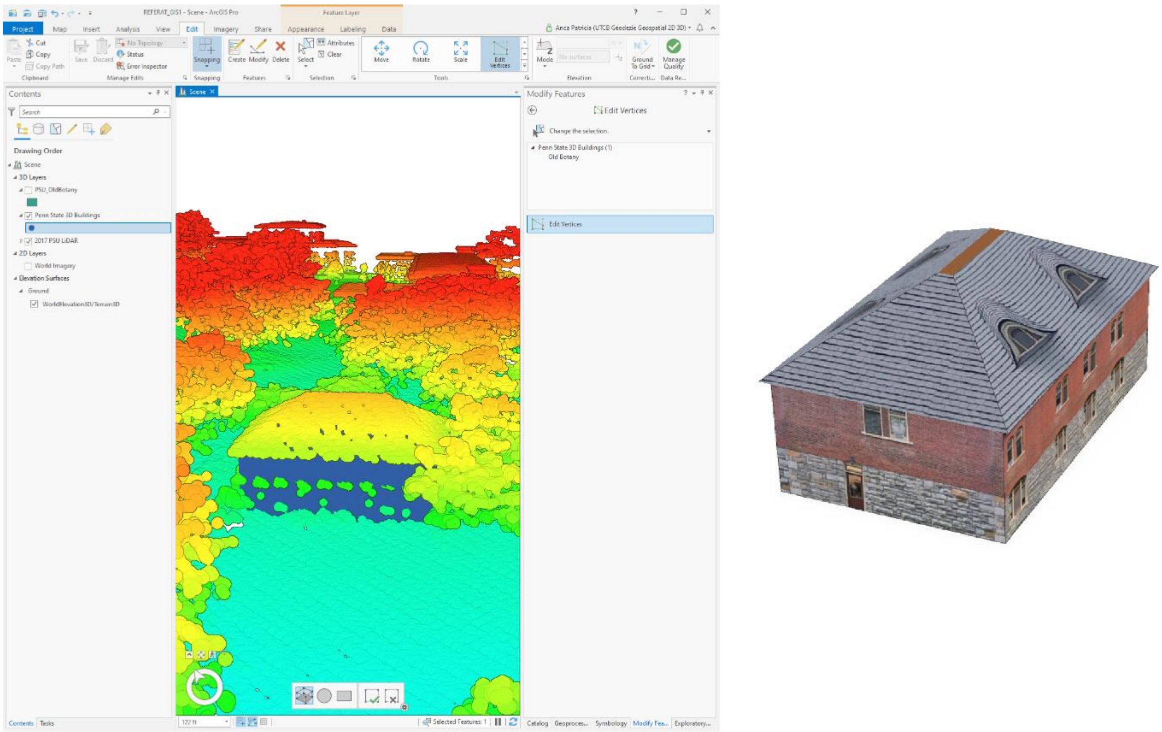


Figure 2.18. Editing Buildings in ArcGIS Pro (Model made during the Geographic Information Systems course, Technical University of Civil Engineering Bucharest, Doctoral School)

In the modelling process, both geospatial data, obtained by the user through different acquisition techniques and data existing in the online catalogue – Living Atlas – which includes different data, maps or applications created by ArcGIS users around the world, can be used. Available data can be photogrammetric images, basic maps, demographics, administrative boundaries, transport lines, urban systems, historical maps, or observations of the Earth (earthquakes, fires, weather, etc.) (The ArcGIS Book | The ArcGIS Book).

For cases where the data retrieved are not complete, missing values may be interpolated from data collected from the same area. There are different interpolation methods—for example, the Kriging function—chosen based on available data and modelled objects or phenomena (The ArcGIS Book | The ArcGIS Book).

2.6.3. Exploring the Possibilities for 3D Modelling Based on eTerra 3 Data in ArcGIS Pro – Case Study

The chosen study area is a residential area in the city of Bragadiru (*Figure 2.19*).

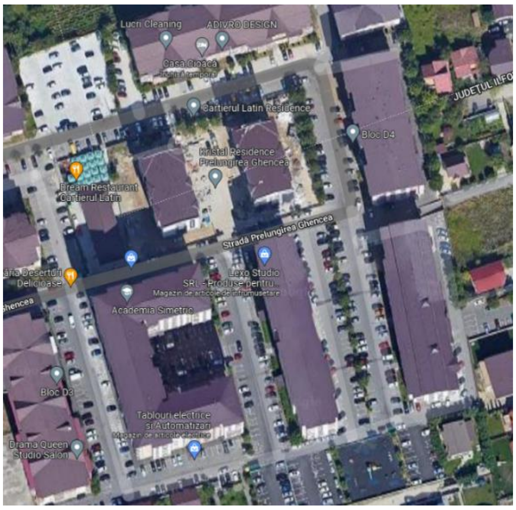


Figure 2.19. Study Area (Google Maps)

The initial data was exported from the eTerra3 application and represents: the buildings footprints, the address and the number of floors of the buildings, depending on which the height was approximated, considering that the height of a floor is 2.75 m.

In order to be able to extrude the buildings footprints, I imported a file in *.shp format, and the created layer was moved to the 3D layer area (*Figure 2.20*). In the attribute table I added the following new fields: Număr_de_niveluri (number of floors), Înălălțime_etaj (floor height), Înălțime_construcție (building height), Adresa (address). After populating the table with data exported from the eTerra3 application, I determined the approximate height of each building with the *Field Calculator* function. The result can be seen in *Figure 2.21*.

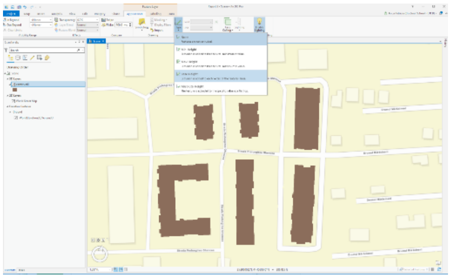


Figure 2.20. Buildings Footprints Imported into ArcGIS Pro

constructii

Field: Add | Calculate | Selection: Select By Attributes | Zoom To | Switch | Clear | Delete | Copy

FID	Shape	LAYER	GM_TYPE	Numar_de_niveluri	Inaltime_constructie	Inaltime_etaj	Adresa
0	Polygon	CONSTRUCTII_ACTIVE	Unknown Area Type	8	22	2.75	Loc. Bragadiru, Str Prelungirea Ghencea, Nr. 45, Bl. C 3
1	Polygon	CONSTRUCTII_ACTIVE	Unknown Area Type	9	24.75	2.75	Loc. Bragadiru, Str Prelungirea Ghencea, Nr. 45, Bl. D5
2	Polygon	CONSTRUCTII_ACTIVE	Unknown Area Type	10	27.5	2.75	Loc. Bragadiru, Str Prelungirea Ghencea, Nr. 45, Bl. D4
3	Polygon	CONSTRUCTII_ACTIVE	Unknown Area Type	11	30.25	2.75	Loc. Bragadiru, Str Prelungirea Ghencea, Nr. 45, Bl. D3
4	Polygon	CONSTRUCTII_ACTIVE	Unknown Area Type	8	22	2.75	Loc. Bragadiru, Str Prelungirea Ghencea, Nr. 45, Bl. C4, Sc. 2
5	Polygon	CONSTRUCTII_ACTIVE	Unknown Area Type	8	22	2.75	Loc. Bragadiru, Str Prelungirea Ghencea, Nr. 45, Bl. C4, Sc. 2

Click to add new row.

Figure 2.21. Table of Attributes of the Layer with the Buildings Footprints

For the extrusion of building geometries, I used the *Extrusion* function, choosing the *Înălțime_construcție* field. The symbology modification was made by choosing the *stacked* symbol, which divides the texture of the buildings by the number of floors according to the corresponding height, and the result can be seen in *Figure 2.22*.

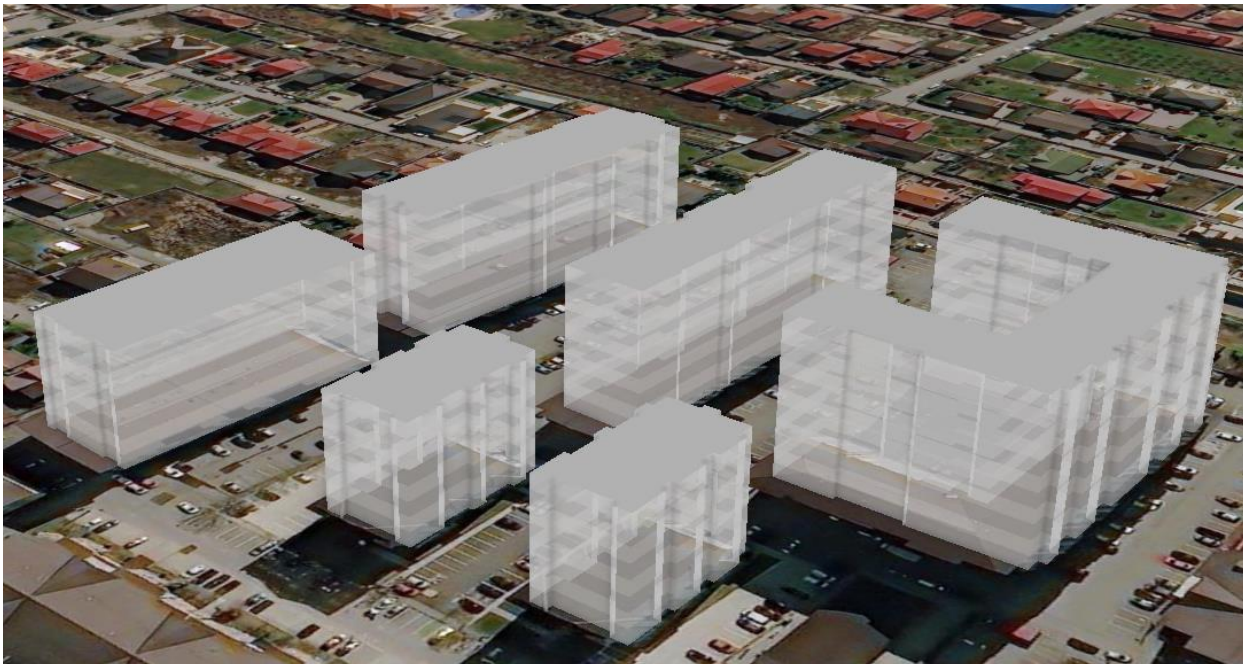


Figure 2.22. The Result of Buildings Footprints Extrusion

ArcGIS Pro also allows adding bookmarks, in order to facilitate map navigation (*Figure 2.23*).

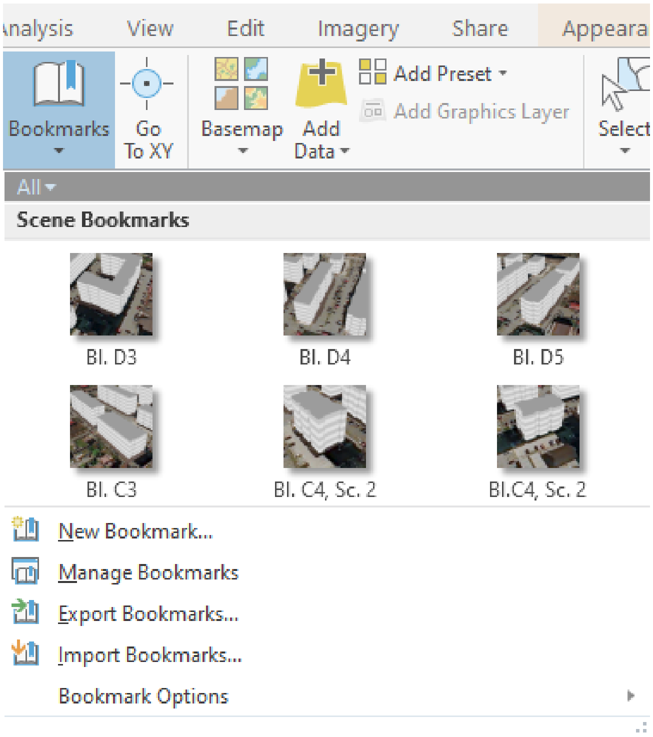


Figure 2.23. Bookmarks in ArcGIS Pro

2.7. Using CAD Software Products for 3D Spatial Modelling

2.7.1. Autodesk 3ds Max Software Product

Autodesk 3ds Max solution is a modelling software product that is suitable both for 3D spatial modelling of the buildings (*Figure 2.24*), their structures and interior elements, as well as for making 3D models from animations or video games (3ds Max Software | Get Prices & Buy Official 3ds Max 2022).

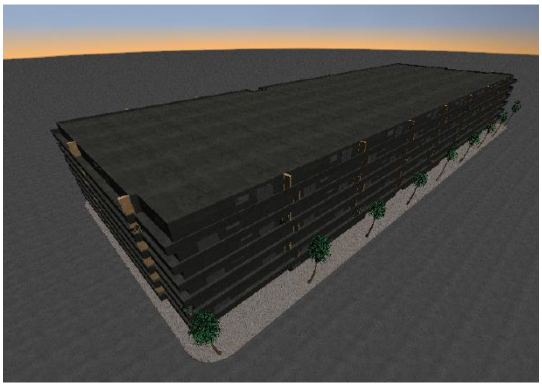


Figure 2.24. 3D Model of a Building, Made in Autodesk 3ds Max Software Product (Autodesk, 2008)

To model different objects, one can start from primitives implemented in the software, or use 2D shapes, generally spleen type, which can be extruded to obtain volumes (Autodesk, 2008).

The *Building Object Style Editor* function allows the generation of 3D models starting from the buildings. A certain style of building can be established for a multitude of buildings by associating a label (shape label masks). From the *Edit Style Element* window (*Figure 2.25*) you can change attributes such as the name of the created element, *Pick Shape Label*, *Building Height*, *Number of Floors*, *Floor Height*, *Wall Mapping Style* (sets how texture is applied), *Wall Material ID* (sets the material identifier associated with walls) (3ds Max 2022 Help | Building Object Style Editor | Autodesk).

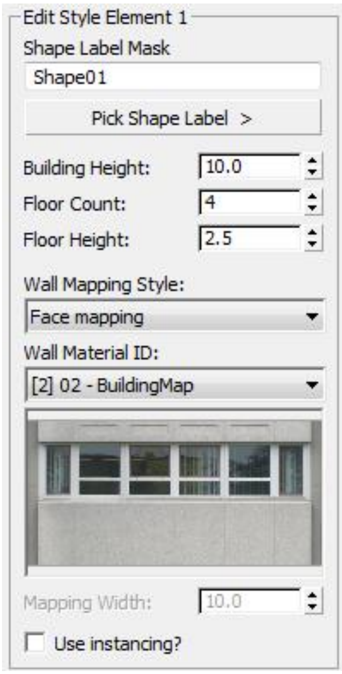


Figure 2.25. Edit Style Element Window in Autodesk 3ds Max (3ds Max 2022 Help | Building Object Style Editor | Autodesk)

2.7.2. Autodesk Revit Software Product

Revit is an Autodesk BIM (Building Informational Model) product, used to accurately generate spatial models of buildings (*Figure 2.26*), in fields such as architectural design, civil engineering, installations, etc. (Revit Software | Get Prices & Buy Official Revit 2022 | Autodesk).

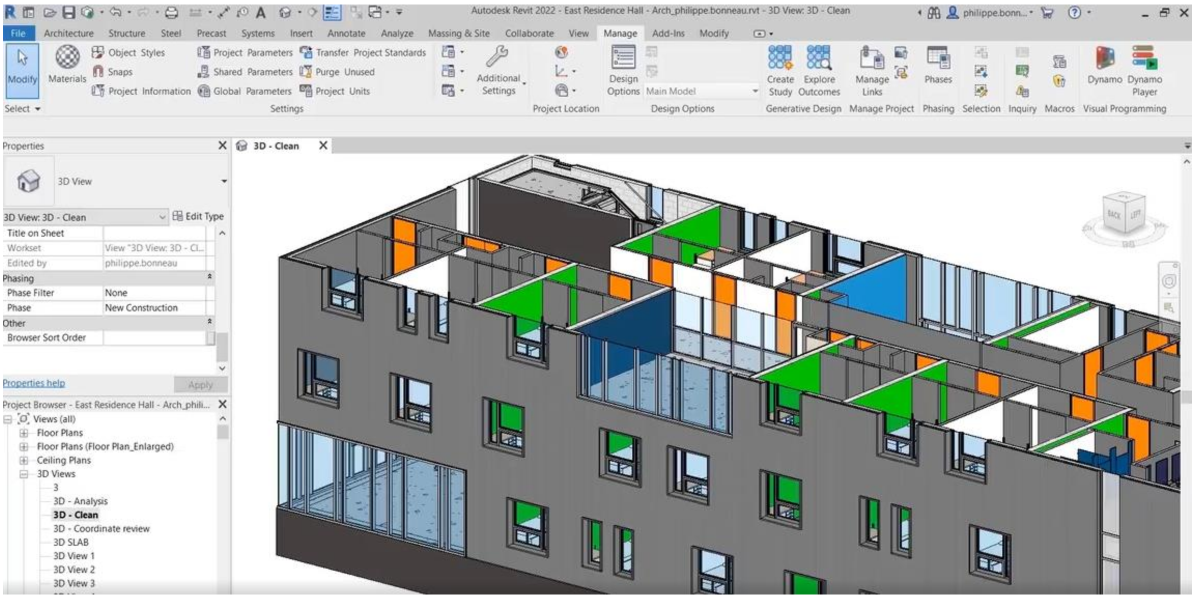


Figure 2.26. 3D Model of a Building made in Autodesk Revit (Revit Software | Get Prices & Buy Official Revit 2022 | Autodesk)

Autodesk Revit software specializes in architectural design and has various design options that allow different possible ways for buildings to be explored in their 3D version (Revit Software | Get Prices & Buy Official Revit 2022 | Autodesk).

2.7.3. Autodesk Civil 3D Software Product

Autodesk Civil 3D is a BIM solution with features that facilitate design and documentation. The initial data can be both geometries represented by closed polylines and point clouds obtained from laser scanning. Modelling results can be exported to ArcGIS-compatible formats.

Surface modelling can be done by combinations of point elements, lines and contours, resulting in different types of surfaces such as TIN surfaces - Triangular Irregular Networks, formed by triangulation (*Figure 2.27-b.*) or GRID surfaces, formed by points that make up a network (*Figure 2.27-a.*) (Civil 3D Features | 2022, 2021 Features | Autodesk).

You can also use the Autodesk Dynamo for Civil 3D extension, which allows you to automate the modelling process by creating a set of rules that will be written in a Visual Programming Language (VLS) and applied (About | The Dynamo Primer).

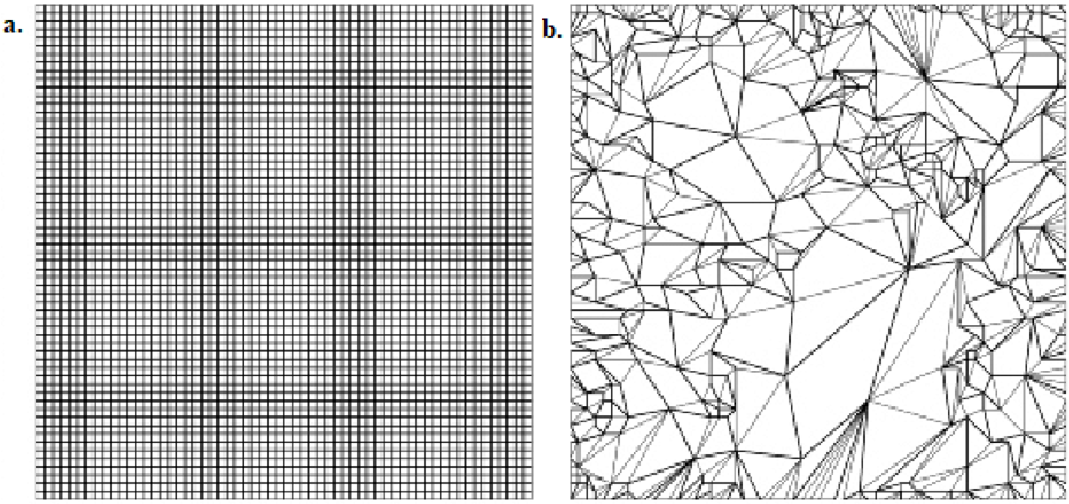


Figure 2.27. Type a. GRID and b. TIN Surfaces (Rahman et al., 2010)

3. 3D Spatial Modelling Based on Photogrammetric Images

3.1. Introduction

Image classification (*Figure 3.1*) involves categorizing and labelling groups of pixels or vectors in an image, based on a set of specific rules. Classification rules may be designed based on one or more spectral or textural characteristics. The main classification methods are: supervised classification and unsupervised classification (Gavali & Banu, 2019).

Unsupervised classification is performed in a fully automatic way, through specific algorithms, while supervised classification involves manually selecting samples (training datasets) from the image and assigning them to predetermined categories, such as roads, buildings, water bodies, vegetation, etc., leading to statistical measures that are applied to the entire image (Gavali & Banu, 2019).

There are different techniques for classifying images, such as classification using neural networks, Support Vector Machine or Random Forest algorithms, Fuzzy logic, genetic algorithms, etc. (Gavali & Banu, 2019).

Various factors can affect the results of the image classification process, such as the choice of classification method, the availability of specific software products, but also the complexity of the terrain in the study area. An essential step in image classification is the pre-processing of the initial data, which may include detection and recovery of distorted lines, geometric rectification, radiometric calibration, application of atmospheric correction and topographic corrections (Lu, 2007).

Currently, the main classification methods are based on machine learning techniques, an application based on artificial intelligence (AI), which includes algorithms that analyze data, learn from it and apply what they have learned in decision making (Deep Learning vs. Machine Learning, 2020).

Figure 3.1. Orthorectified Image (left) and Image Classification Result (right): red-construction, dark green-tree, light green-low vegetation, grey-waterproof surface, yellow-cars (Höhle, 2017)

3.2. Objects Extraction from Images using Machine Learning

3.2.1. *Support-Vector Machine* Algorithm

Support Vector Machine (SVM) is one of the most widely used algorithms for supervised image classification. The quality of the results of the SVM classification process depends on the accuracy of the classification, derived from the samples chosen at the training stage and the coverage provided by them. Data redundancy can slow down the training process, by using more memory than necessary (Aslani & Seipel, 2020).

The SVM method involves the use of a training dataset, based on which the algorithms will divide the dataset into several classes, according to the initially chosen samples. The data will thus be separated by optimal separation hyperplanes, which are, in fact, the decision limit between the characteristics of different classes (Maulik & Chakraborty, 2017).

In *Figure 3.2*, the blue dots and red triangles represent two classes of elements, represented by the characteristics x and y, and the black line represents the decision boundary (a hyperplane represented in 2D space). The SVM algorithm takes the data and creates the most appropriate boundary that separates it into two distinct categories.

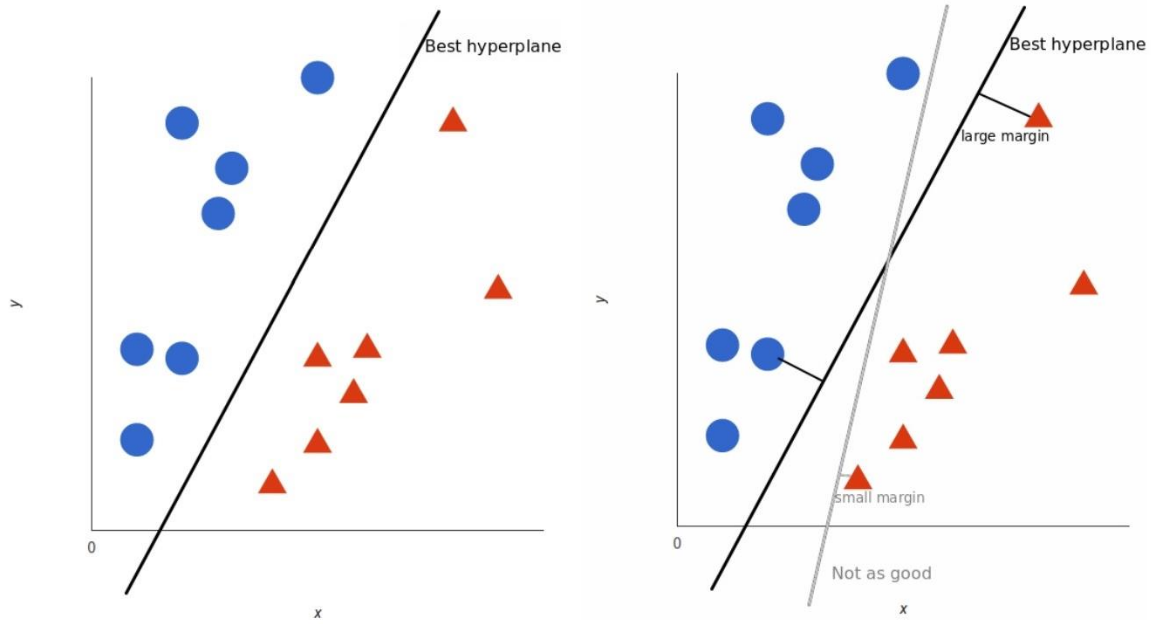


Figure 3.2. Support Vector Machine (SVM) Algorithm Explained, 2017)

When using the SVM algorithm, the optimal hyperplane is considered to be the one that takes into account the maximum distance from the class characteristics, i.e. the highest limit (Support Vector Machines (SVM) Algorithm Explained, 2017).

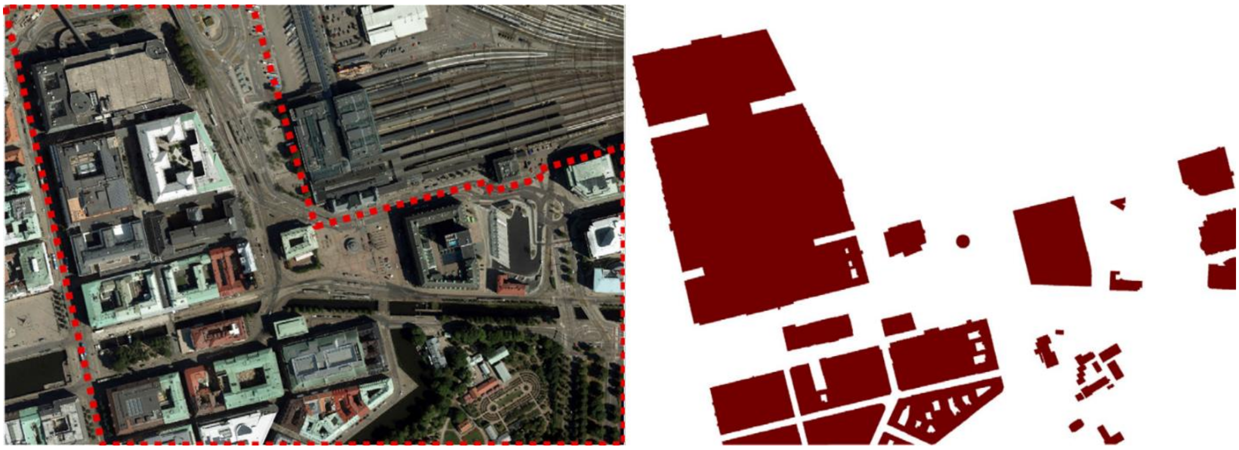


Figure 3.3. Classification of an Image (left side) by SVM Algorithm, Obtaining the Buildings Footprints (right side) (Aslani &; Seipel, 2020)

The initial data used to obtain building geometries by classifying images with the SVM algorithm (*Figure 3.3*) can be orthorectified images, LiDAR data, buildings footprints, point clouds derived from oblique photogrammetric images with Dense Image Matching, etc. (Aslani & Seipel, 2020).

3.2.2. *Random Forest* Algorithm

The *Random Forest* classification algorithm is a supervised classification algorithm that is based on a combination of decision trees. It selects random data from the original dataset, generates a decision tree for each sample, and each tree casts a unified vote for the most popular class (Breiman, 1999). Finally, for assigning a pixel to a given class, the algorithm selects the most voted result as the final prediction (*Figure 3.4*).

In the training process, trees are created by extracting a subset of training data, by substitution; thus, a sample may be selected several times, and others may not be selected at all. Around two-thirds of the samples are used in the training process, and the rest are used to estimate classification performance (Belgiu & Drăguţ, 2016).

Two parameters are required to generate the trees: the number of decision trees (*Ntree*) and the number of variables to be selected and tested for tree generation (*Mtry*). The number of trees can be as large as necessary, but studies have shown that at a number of around 500, the classification errors will stabilize. The number of variables is usually established as the square root of the number of input variables (Belgiu &; Drăguţ, 2016).

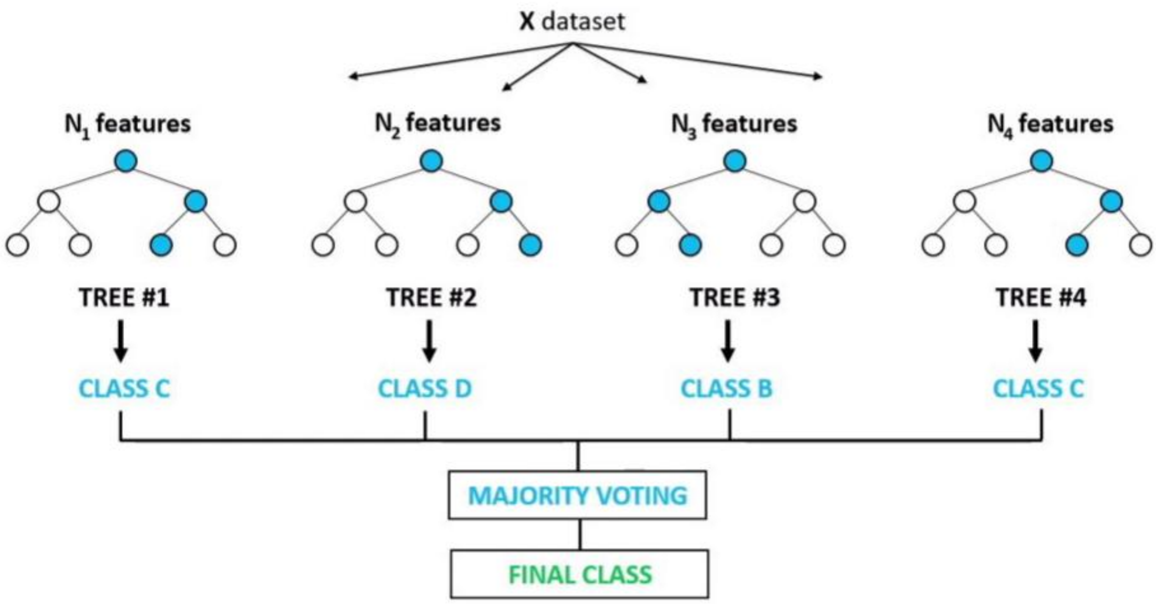


Figure 3.4. Classification with Random Forest Algorithm (Chauhan, 2021)

In the semantic classification of images for extracting the geometry of buildings (*Figure 3.5*), the *Random Forest* algorithm is distinguished by its ability to use a large volume of sampling data.

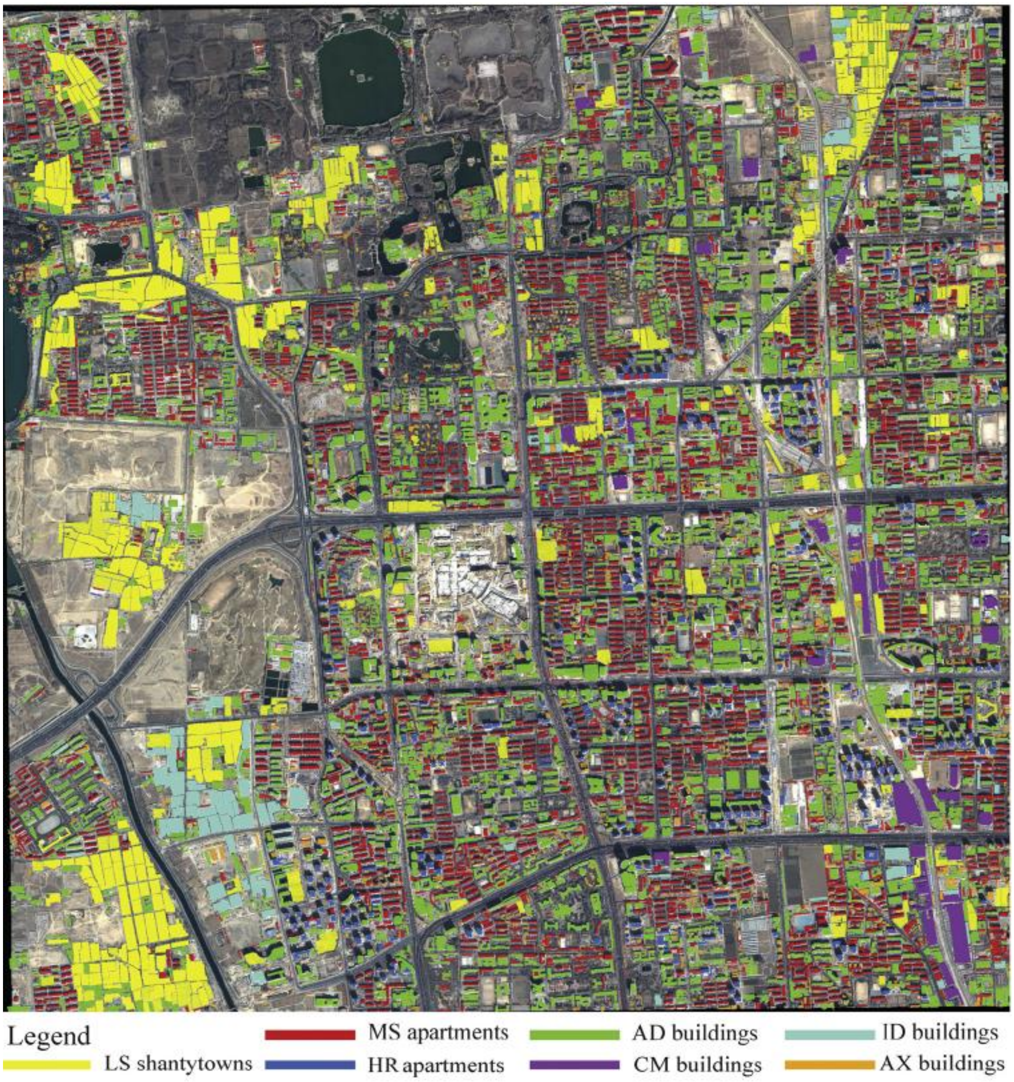


Figure 3.5. Results of a Classification with the Random Forest Algorithm (Du et al., 2015)

3.3. Extracting Objects from Images Using Artificial Neural Networks

3.3.1. Introduction

Neural networks have many applications in various fields such as mathematics, physics, statistics, or engineering, being used to model phenomena. An artificial neural network is an information processing system, similar to a biological neural network, which has as its main component an artificial neuron similar to a biological neuron and has the following characteristics (Yadav, 2015):

- Information processing is done through simple connections called neurons (*Figure 3.6*);
- Signals are transferred between neurons via connecting links;
- Each connection link is assigned a certain strength, which multiplies the transmitted signal;
- An activation function is applied to the net input signal by each neuron to determine its output signal.

Each neuron receives one or more input signals through connections also called synapses, and produces a single output signal (Yadav, 2015).

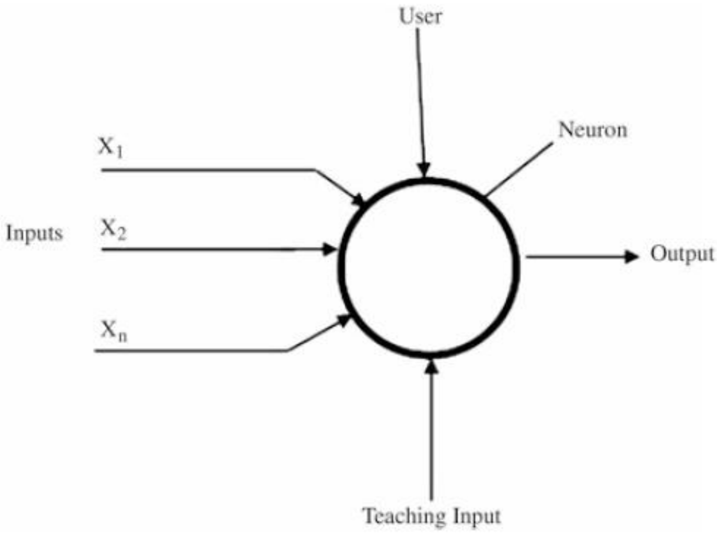
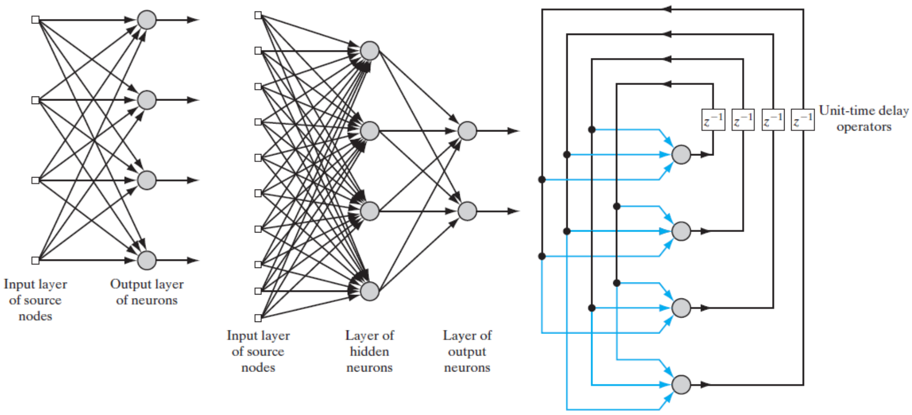


Figure 3.6. The Structure of an Artificial Neuron (Yadav, 2015)

3.3.2. Neural Networks Architecture

The structure of neural networks is given by the algorithm used in the training process. There are three types of neural network architectures (Haykin, 2008):

- *One-layer Feedforward networks* (*Figure 3.7.a*): represents the simplest form of a neural network and involves an input layer of source nodes, which is projected into an output layer of neurons, but not vice versa;
- *Multi-layer Feedforward networks* (*Figure 3.7.b*): are characterized by the presence of one or more hidden layers, whose computational nodes are referred to as hidden neurons or hidden units;
- *Recurrent networks* (*Figure 3.7.c*): unlike Feedforward networks, they are characterised by at least one feedback loop.



a. *b.* *c.*

Figure 3.7. Types of Neural Network Architectures (Haykin, 2008)

3.3.3. Convolutional Neural Networks

The term Deep Learning refers to multi-layered artificial neural networks. One of the most widely used deep artificial neural networks is the Convolutional Neural Network (CNN). An important feature of this type of network is that the determinations made are not dependent on the position in space of objects, meaning they can be identified regardless of their position in the image (Albawi et al., 2017).

Deep learning methods have proven to be very effective in the process of semantic segmentation of images, with the emerge of FCN (Fully Convolutional Networks) neural networks, characterized by a small number of parameters that allow reducing the time required for the training process (Long et al., 2015).

SegNet is an FCN neural network based on an encoding-decoding architecture (*Figure 3.8*) that generates output data with the same resolution as the input data.

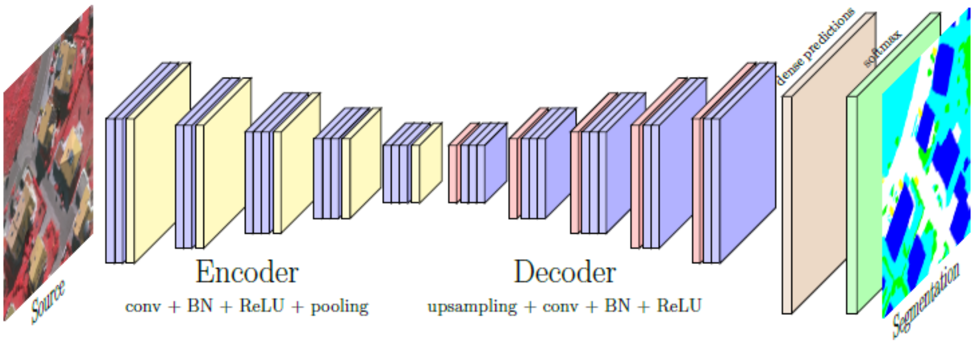


Figure 3.8. SegNet Neural Network Architecture (Audebert et al., 2017)

Figure 3.9 shows an example of a classification using the SegNet neural network, in which roads are represented in white, buildings are shown in blue, in cyan – low vegetation, in green – trees and in yellow – cars.

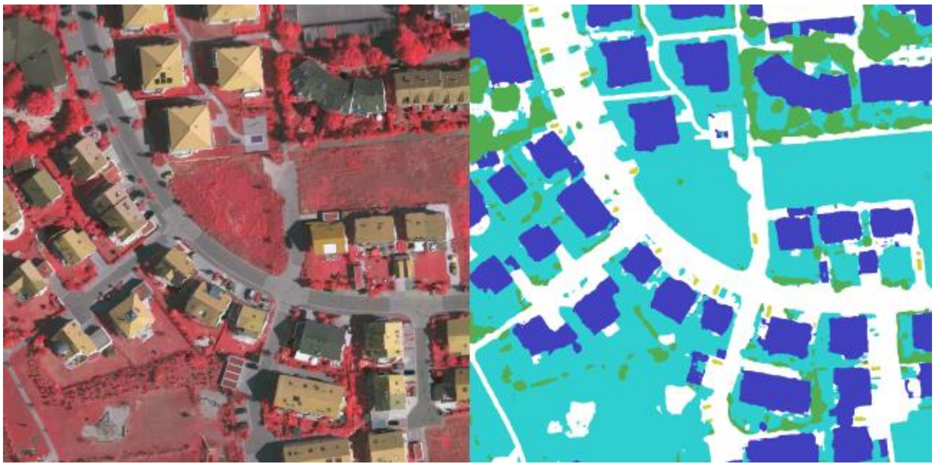


Figure 3.9. Image (left-false colour), Classified (right) using the SegNet Neural Network (Audebert et al., 2017)

3.4. Obtaining 3D Models of the Buildings from Buildings Footprints Extracted with a Deep Learning Algorithm in ArcGIS Pro – Case Study

ArcGIS Pro software includes tools that use AI and deep learning methods to extract features from images. This requires the *Deep Learning Libraries Installers* to be installed in ArcGIS Pro (*Deep Learning Libraries Installers for ArcGIS*, 2020/2022).

In Living Atlas there are also different models for buildings footprint extraction based on satellite images. In this case, I used the *Building Footprint Extraction – USA* model, based on a convolutional neural network with a Mask R-CNN architecture (Building Footprint Extraction - USA - Overview).

The study area is a residential neighbourhood in Craiova (*Figure 3.10*).

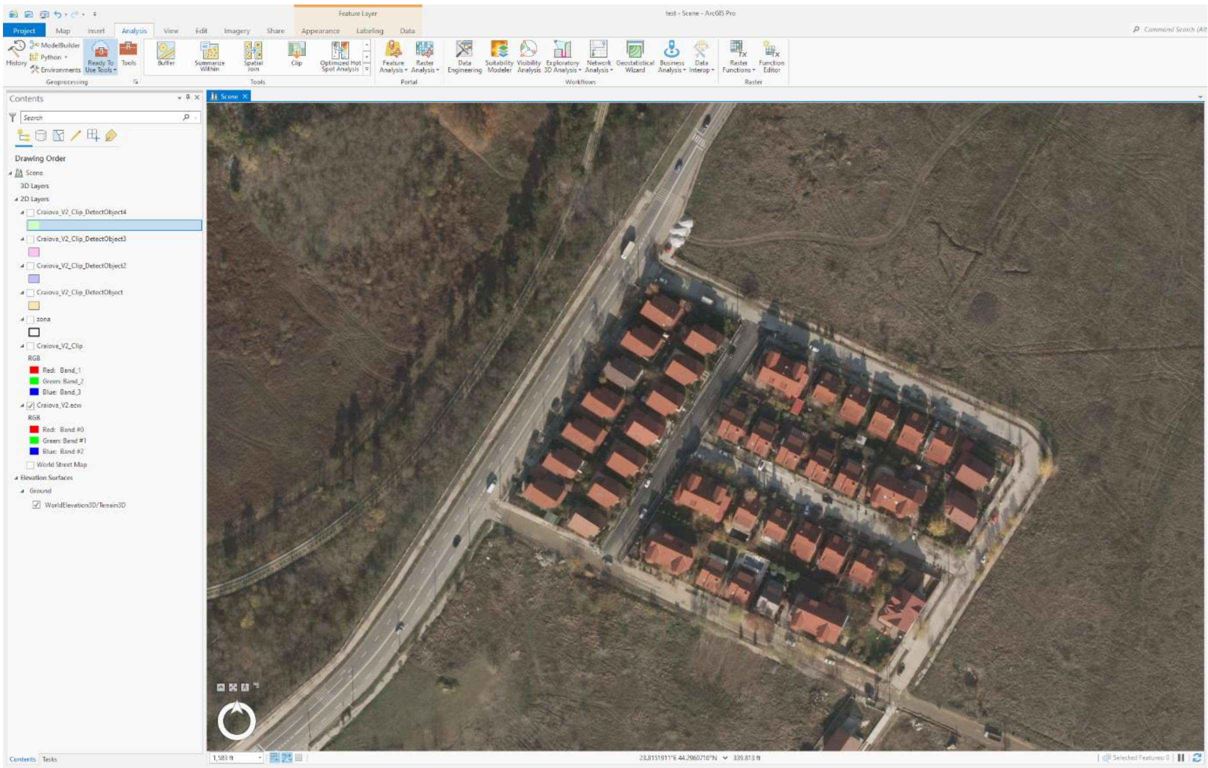


Figure 3.10. Study Area

The *Detect Objects Using Deep Learning* geoprocessing tool applies an indicated deep learning model to a raster image, resulting in a new class of features containing the objects identified in the image.

To set the parameters (*Figure 3.11*), as input layer I chose the layer with the raster image, and the output layer would be the layer containing the buildings footprints extracted from the image. The coordinate system was set to be chosen from the created Scene, i.e. the Stereo 70 system, and for the processing extent I chose the current screen extension. The function also allows you to choose the type of processor used for image segmentation, namely: CPU (central processing unit) or GPU (graphics processing unit).

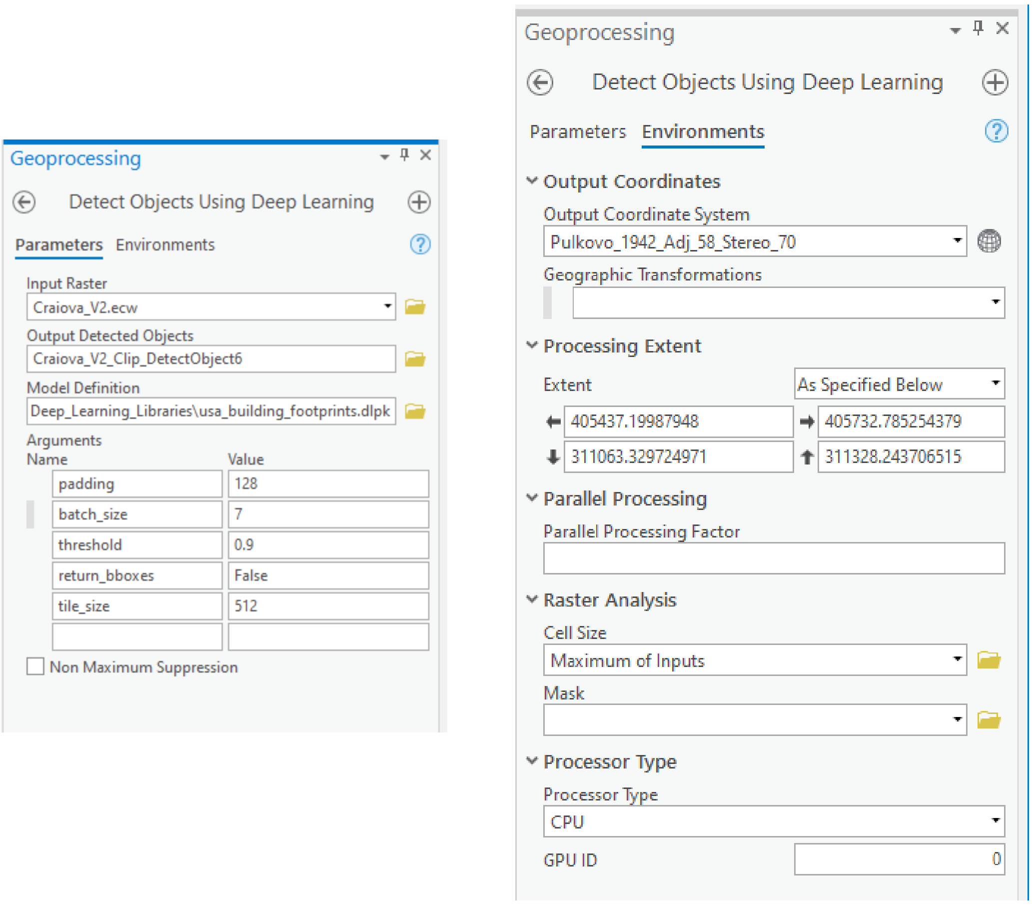


Figure 3.11. Setting the Parameters for the Detect Objects Using Deep Learning Function

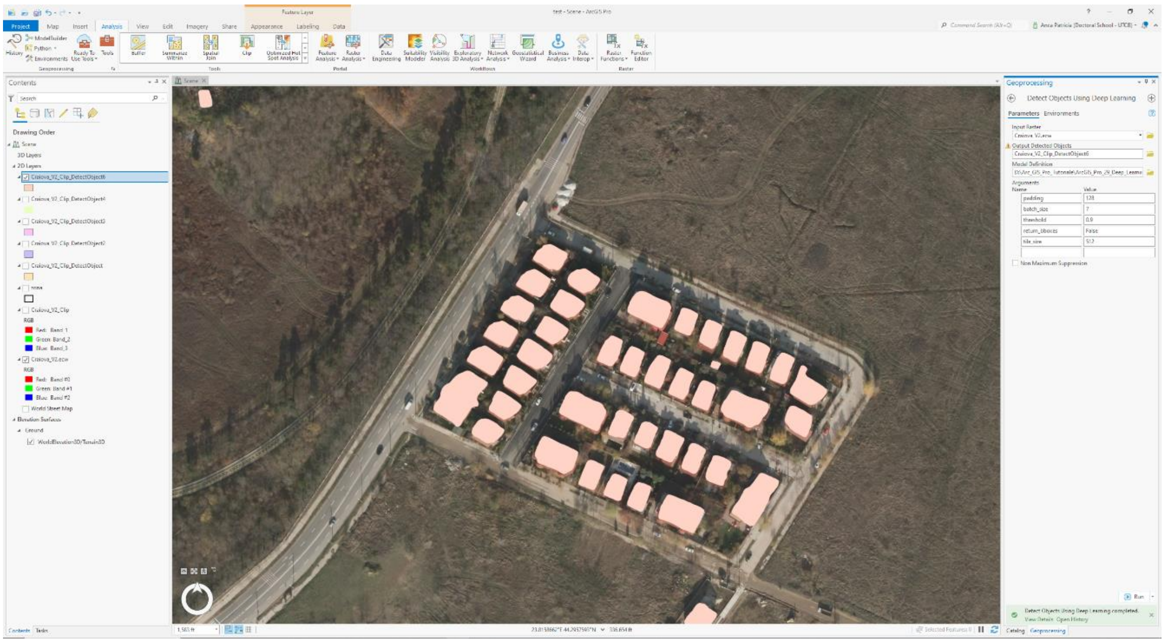


Figure 3.12. Image Segmentation Result with Detect Objects Using Deep Learning in ArcGIS Pro

The result of image segmentation can be seen in *Figure 3.12*. The buildings footprints extracted from the image are not regular, and to regularize them I used the *Regularize* (*Figure 3.13*) geoprocessing tool, choosing as input layer the previously obtained layer, as output layer the new layer with the regularized buildings footprints, and the *Right Angles* method, to obtain geometries with right angles.

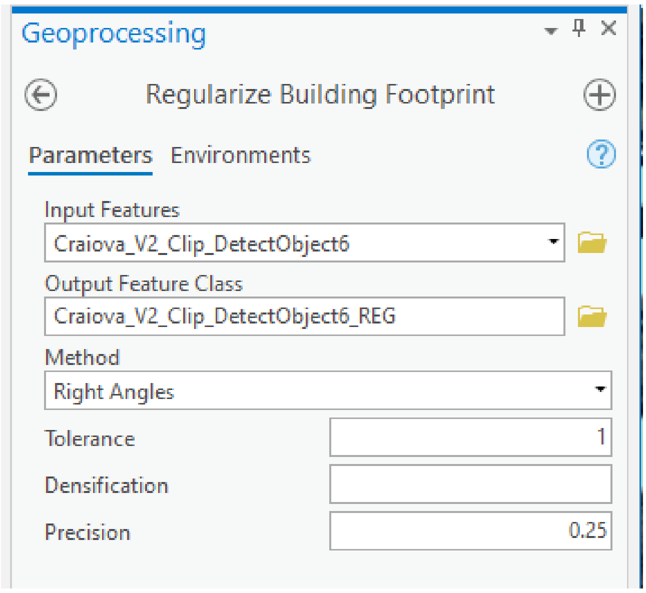


Figure 3.13. Setting the Parameters for the Regularize Function

The final result of extracting buildings footprints from satellite imagery using the deep learning algorithm implemented in ArcGIS Pro can be seen in *Figure 3.14*.

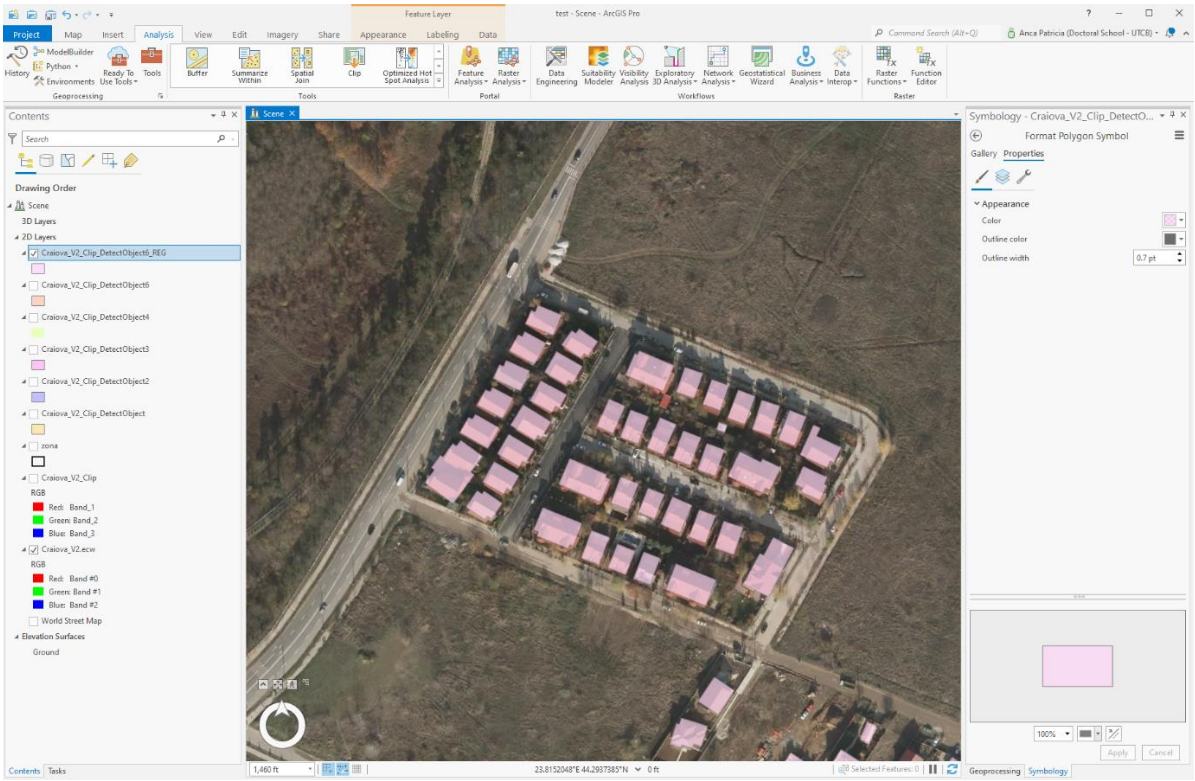


Figure 3.14. The Result of Extracting Buildings Footprints from Satellite Imagery Using the Deep Learning Algorithm Implemented in ArcGIS Pro

Based on the floors number of the buildings, their height was estimated (*Figure 3.15*).

Craiova_V2_Clip_DetectObject6_REG

Field: Add Calculate Selection: Select By Attributes Zoom To Switch Clear Delete Copy

	OID *	Class	Confidence	Shape *	ORIG_OID	STATUS	Shape_Length	Shape_Area	Numar_niveluri	Inaltime_constructie	Inaltime_etaj
2	2	1	98.450643	Polygon	2	0	50.204581	146.25	3	7.5	2.5
3	3	1	97.37879	Polygon	3	0	50.204581	147	3	7.5	2.5
4	4	1	96.599132	Polygon	4	0	49.497475	145	3	7.5	2.5
5	5	1	98.201215	Polygon	5	0	52.325902	163	3	7.5	2.5
6	6	1	97.797024	Polygon	6	0	50.911688	154.875	3	7.5	2.5
7	7	1	97.222275	Polygon	7	0	50.204581	147.5	3	7.5	2.5

Figure 3.15. Table of Attributes of the Buildings Layer

With the *Extrusion* function, the buildings footprints were extruded, depending on their height, and the result can be seen in *Figure 3.16*.

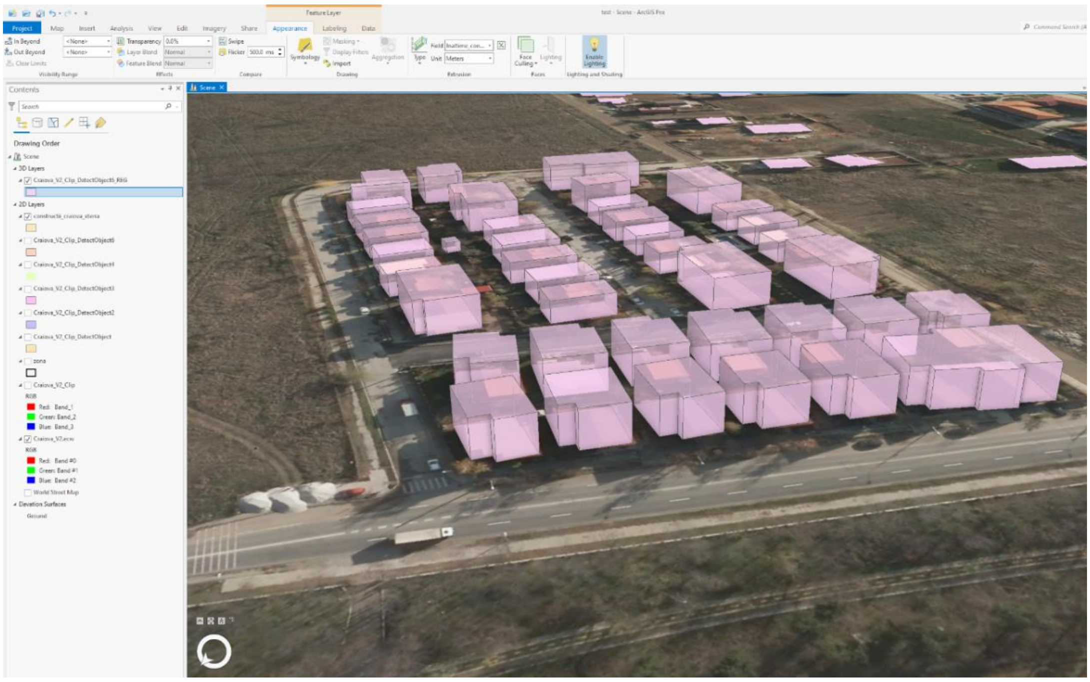


Figure 3.16. Extrusion of Buildings Footprints

4. Conclusions

The purpose of this research report was to analyze the main methods and software products for 3D spatial modelling of the buildings, in the context of an increasing need for the development of 3D models of cities.

The data sources needed to generate 3D models of the cities are varied and can be point clouds, orthorectified images, buildings footprints, elevation data, etc. Depending on the availability of this data, we can choose an optimal modelling method and software. From photogrammetric images, building geometries can be obtained through various machine learning and deep learning processes, while point clouds, buildings footprints and other elevation data can be the basis of 3D models made with GIS or CAD software products.

CityGML standards create a basis for numerous 3D modelling applications, ensuring interoperability between them. CityGML comprises five levels of detail, which define the completeness of the data in the composition of the 3D model. The model proposed by Esri – 3DCIM (3D City Information Model) is a geospatial database that provides data management and allows data exchange between it and the CityGML model, the two being interoperable. Geospatial data is associated with attributes, together forming a semantic model of the city.

Software products such as Autodesk 3ds Max, Revit and Civil 3D are generally suitable for design or BIM applications, but their results can be used to complement existing data and improve 3D models of the cities.

The supported format of the data is also an important factor, ensuring the integration of data into several types of applications. Thus, the described methods and software products allow importing and exporting data in several types of formats, such as .dxf, .shp, .ascii, .xml, .gdb, .tiff, etc..

The automation of the modelling process is a very important factor, especially when working with a large volume of data. In case of using the methods for generating 3D spatial models based on point clouds, we can talk about model-driven methods, which use primitives described in a library, or data-driven methods, which analyze the point cloud in its entirety, without dividing the building into primitives. When generating 3D models based on orthorectified images, I identified different machine learning algorithms, such as the *Support Vector Machine* algorithm or the *Random Forest* algorithm, used to classify images. In addition, another method used to classify images is the use of artificial neural networks, which reduce the time required for the entire process.

ArcGIS Pro software has distinguished itself by incorporating a number of 3D modelling possibilities from different data sources, including 3D point clouds and photogrammetric images.

From the case studies it is noted that in order to obtain a higher level of detail, it is necessary to acquire spatial data through laser scanning techniques, the point cloud offering the possibility of modelling details such as roofs and terraces. Existing building data in the eTerra3 application can help generate 3D models with a lower level of detail, but they do not provide sufficient coverage.

Following the study, I propose to consider creating, at national level, an open database containing different types of attributes related to buildings such as the year of construction, number of levels, height, building destination, area, address, or other relevant data to which the general public can have access. This database can be useful in areas such as solar radiation estimation, energy demand estimation, positioning, building classification, improvement of visualization methods, visibility analysis, estimation of shadow areas produced by buildings, noise propagation estimation, urban planning, emergency response, population estimation, disaster prediction, archaeology, etc..

In conclusion, taking into account the initial available data, or the possibilities of their acquisition, the level of detail imposed by the purpose of the project and also the type of application that requires 3D modelling of buildings, optimal methods and software products can be chosen to obtain 3D spatial models of buildings in accordance with the accuracy requirements of the project.

References

1. *2D, 3D & 4D GIS Mapping Software | ArcGIS Pro*. Retrieved on 9 march 2022, from https://www.esri.com/en-us/arcgis/products/arcgis-pro/overview
2. *3ds Max 2022 Help | Building Object Style Editor | Autodesk*. Retrieved on 29 march 2022, from https://help.autodesk.com/view/3DSMAX/2022/ENU/?guid=GUID-6D823C77-D233-4FD4-89BC-FAE2F7EAA785
3. *3ds Max Software | Get Prices & Buy Official 3ds Max 2022*. Retrieved on 29 march 2022, from https://www.autodesk.com/products/3ds-max/overview
4. *About | The Dynamo Primer*. Retrieved on 4 april 2022, from https://primer.dynamobim.org/en/
5. *About CityEngine—ArcGIS CityEngine Resources | Documentation*. Retrieved on 22 february 2022, from https://doc.arcgis.com/en/cityengine/latest/get-started/get-started-about-cityengine.htm
6. *Advanced 3D City Design Software | ArcGIS CityEngine*. Retrieved on 22 february 2022, from https://www.esri.com/en-us/arcgis/products/arcgis-cityengine/overview
7. *ArcGIS Pro Resources | Tutorials, Documentation, Videos & More*. Retrieved on 9 march 2022, from https://www.esri.com/en-us/arcgis/products/arcgis-pro/resources
8. Aslani, M., & Seipel, S. (2020). A fast instance selection method for support vector machines in building extraction. *Applied Soft Computing*, *97*, 106716. https://doi.org/10.1016/j.asoc.2020.106716
9. Audebert, N., Saux, B., & Lefèvre, S. (2017). Beyond RGB: Very High Resolution Urban Remote Sensing With Multimodal Deep Networks. *ISPRS Journal of Photogrammetry and Remote Sensing*, *140*. https://doi.org/10.1016/j.isprsjprs.2017.11.011
10. Autodesk. (2008). Understanding 3ds Max. În Autodesk (Ed.), *Learning Autodesk 3ds Max 2008 Foundation* (pp. 13–37). Focal Press. https://doi.org/10.1016/B978-0-240-80927-4.50010-0
11. Badea, A. C. (2020). *Note de curs, disciplina "Sisteme Informatice Geografice"*. Universitatea Tehnică de Construcții București, Școala Doctorală.
12. Belgiu, M., & Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing*, *114*, 24–31. https://doi.org/10.1016/j.isprsjprs.2016.01.011
13. Biljecki, F., Ledoux, H., & Stoter, J. (2016). An improved LOD specification for 3D building models. *Computers, Environment and Urban Systems*, *59*, 25–37. https://doi.org/10.1016/j.compenvurbsys.2016.04.005

14. Biljecki, F., Ledoux, H., Stoter, J., & Zhao, J. (2014). Formalisation of the level of detail in 3D city modelling. *Computers, Environment and Urban Systems*, *48*, 1–15. https://doi.org/10.1016/j.compenvurbsys.2014.05.004

15. Biljecki, F., Lim, J., Crawford, J., Moraru, D., Tauscher, H., Konde, A., Adouane, K., Lawrence, S., Janssen, P., & Stouffs, R. (2021). Extending CityGML for IFC-sourced 3D city models. *Automation in Construction*, *121*, 103440. https://doi.org/10.1016/j.autcon.2020.103440

16. Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., & Çöltekin, A. (2015). Applications of 3D City Models: State of the Art Review. *ISPRS International Journal of Geo-Information*, *4*(4), 2842–2889. https://doi.org/10.3390/ijgi4042842

17. Billen, R., Cutting-Decelle, A., Marina, O., Almeida, J.-P., Caglioni, M., Falquet, G., Leduc, T., Métral, C., Moreau, G., Perret, J., Rabino, G., García, R., Yatskiv, I., & Zlatanova, S. (2014). *3D City Models and urban information: Current issues and perspectives*. https://doi.org/10.1051/TU0801/201400001

18. Breiman, L. (1999). *1 RANDOM FORESTS--RANDOM FEATURES*. http://citeseerx.ist.psu.edu/viewdoc/citations;jsessionid=0822AD486441F0148CAFA1C643FF8F20?doi=10.1.1.367.9714

19. *Building Footprint Extraction—USA - Overview*. Retrieved on 12 april 2022, from https://sd-utcb.maps.arcgis.com/home/item.html?id=a6857359a1cd44839781a4f113cd5934

20. Chauhan, A. (2021, februarie 23). Random Forest Classifier and its Hyperparameters. *Analytics Vidhya*. https://medium.com/analytics-vidhya/random-forest-classifier-and-its-hyperparameters-8467bec755f6

21. *CityEngine Web Viewer—Kisumu3D*. Retrieved on 22 february 2022, from https://www.arcgis.com/apps/CEWebViewer/viewer.html?&3dWebScene=e2981d2427eb4be28ecb7270adb6fcb3&view=694978.69,6.1,-9988257,694117.5,899.59,-9987554,0.5&lyr=1,1,1,1,0,0,1,1,1,1,1,1,0,0,0,0,0,0&wkid=32736&v=2

22. *CityGML | OGC*. Retrieved on 18 february 2022, from https://www.ogc.org/standards/citygml

23. *Civil 3D Features | 2022, 2021 Features | Autodesk*. Retrieved on 29 march 2022, from https://www.autodesk.com/products/civil-3d/features

24. *Deep Learning Libraries Installers for ArcGIS*. (2022). Esri. https://github.com/Esri/deep-learning-frameworks/blob/f7c3d45b195e42bead5eba0caf41d7507abbf511/README.md?rmedium=links_esri_com_b_d&rsource=https%3A%2F%2Flinks.esri.com%2Fdeep-learning-framework-install (Original work published 2020)

25. *Deep learning vs. machine learning: What's the difference?* (2020, january 23). Zendesk. https://www.zendesk.com/blog/machine-learning-and-deep-learning/

26. Du, S., Zhang, F., & Zhang, X. (2015). Semantic classification of urban buildings combining VHR image and GIS data: An improved random forest approach. *ISPRS Journal of Photogrammetry and Remote Sensing*, *105*, 107–119. https://doi.org/10.1016/j.isprsjprs.2015.03.011

27. *Essential shape operations overview—ArcGIS CityEngine Resources | Documentation*. Retrieved on 22 february 2022, from https://doc.arcgis.com/en/cityengine/latest/help/help-essential-shape-operations-overview.htm

28. Gavali, P., & Banu, J. S. (2019). Chapter 6—Deep Convolutional Neural Network for Image Classification on CUDA Platform. În A. K. Sangaiah (Ed.), *Deep Learning and Parallel Computing Environment for Bioengineering Systems* (pp. 99–122). Academic Press. https://doi.org/10.1016/B978-0-12-816718-2.00013-0

29. *Generation of street networks—ArcGIS CityEngine Resources | Documentation*. Retrieved on 22 februarie 2022, from https://doc.arcgis.com/en/cityengine/latest/help/help-grow-a-street.htm

30. Gröger, G., Kolbe, T., Czerwinski, A., & Nagel, C. (2008). *OpenGIS® City Geography Markup Language (CityGML) Encoding Standard*. Open Geospatial Consortium Inc.

31. Gröger, G., Kolbe, T. H., & Czerwinski, A. (2006). *Candidate OpenGIS® CityGML Implementation Specification (City Geography Markup Language)*. Open Geospatial Consortium Inc.

32. Gröger, G., & Plümer, L. (2012). CityGML – Interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, *71*, 12–33. https://doi.org/10.1016/j.isprsjprs.2012.04.004

33. Haykin, S. (2008). *Neural Networks and Learning Machines*. Prentice Hall.

34. Höhle, J. (2017). Generating Topographic Map Data from Classification Results. *Remote Sensing*, *9*, 224. https://doi.org/10.3390/rs9030224

35. Kolbe, T., Gröger, G., & Plümer, L. (2005). CityGML - Interoperable access to 3D city models. *Geo-information for Disaster Management*. https://doi.org/10.1007/3-540-27468-5_63

36. Kutzner, T., Chaturvedi, K., & Kolbe, T. H. (2020). CityGML 3.0: New Functions Open Up New Applications. *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, *88*(1), 43–61. https://doi.org/10.1007/s41064-020-00095-z

37. Ledoux, H., Biljecki, F., Dukai, B., Kumar, K., Peters, R., Stoter, J., & Commandeur, T. (2021). 3dfier: Automatic reconstruction of 3D city models. *Journal of Open Source Software*, *6*(57), 2866. https://doi.org/10.21105/joss.02866

38. Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3431–3440. https://doi.org/10.1109/CVPR.2015.7298965

39. Lu, D. (2007). A Survey of Image Classification Methods and Techniques for Improving Classification Performance. *International Journal of Remote Sensing*, *28*, 823–870. https://doi.org/10.1080/01431160600746456

40. Maas, H.-G., & Vosselman, G. (1999). Two algorithms for extracting building models from raw laser altimetry data. *ISPRS Journal of Photogrammetry and Remote Sensing*, *54*(2), 153–163. https://doi.org/10.1016/S0924-2716(99)00004-0

41. Maren, G. van. (2014, decembrie 16). Lossless CityGML support in ArcGIS. *ArcGIS Blog*. https://www.esri.com/arcgis-blog/products/3d-gis/3d-gis/lossless-citygml-support-in-arcgis/

42. Maulik, U., & Chakraborty, D. (2017). Remote Sensing Image Classification: A survey of support-vector-machine-based advanced techniques. *IEEE Geoscience and Remote Sensing Magazine*, *5*, 33–52. https://doi.org/10.1109/MGRS.2016.2641240

43. *Overview—3D BAG*. Retrieved on 9 february 2022, from https://docs.3dbag.nl/en/

44. Parish, Y. I. H., & Müller, P. (2001). Procedural modeling of cities. *in Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, 301–308.

45. *Planificare și proiectare urbană—Planificare Urbană Inteligentă | ArcGIS Urban*. Retrieved on 6 april 2022, from https://www.esri.com/ro-ro/arcgis/products/arcgis-urban/overview

46. Rahman, A., Sunar, M. S., Zamri, M. N., & Shaari, N. (2010). *Terrain Rendering on GPU*.

47. Reitz, T., McCabe, C., Morrish, S. W., & Sims, B. (2014). ArcGIS for 3D Cities: An Introduction. *An Introduction*, 56.

48. Reitz, T., & Schubiger-Banz, T. (2014). The Esri 3D city information model. *IOP Conference Series: Earth and Environmental Science*, *18*.

49. *Revit Software | Get Prices & Buy Official Revit 2022 | Autodesk*. Retrieved on 29 march 2022, from https://www.autodesk.com/products/revit/overview

50. S. Albawi, T. A. Mohammed, & S. Al-Zawi. (2017). Understanding of a convolutional neural network. *2017 International Conference on Engineering and Technology (ICET)*, 1–6. https://doi.org/10.1109/ICEngTechnol.2017.8308186

51. Stadler, A., & Kolbe, T. (2012). *Spatio-semantic coherence in the integration of 3D city models*. *36*.

52. *Support Vector Machines (SVM) Algorithm Explained*. (2017, june 22). MonkeyLearn Blog. https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/

53. Takase, Y., Sho, N., Stone, A., & Shimiya, K. (2003). Automatic Generation of 3D City Models and Related Applications. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XXXIV-5/W10*.

54. Tang, L., Ying, S., Li, L., Biljecki, F., Zhu, H., Zhu, Y., Yang, F., & Su, F. (2020). An application-driven LOD modeling paradigm for 3D building models. *ISPRS Journal of Photogrammetry and Remote Sensing*, *161*, 194–207.

55. Tarsha-Kurdi, F., Landes, T., Grussenmeyer, P., & Koehl, M. (2007). Model-driven and data-driven approaches using LIDAR data: Analysis and comparison. *ISPRS Workshop*, 87–92. https://halshs.archives-ouvertes.fr/halshs-00264846

56. *The ArcGIS Book | The ArcGIS Book*. Retrieved on 9 march 2022, from https://learn.arcgis.com/en/arcgis-book/

57. Turksever, S. (2015). *3D MODELING WITH CITY ENGINE*. https://doi.org/10.13140/RG.2.2.30548.30085

58. *Virtual Singapore*. Retrieved on 9 february 2022, from https://www.nrf.gov.sg/programmes/virtual-singapore

59. *Virtual Singapore—Building a 3D-Empowered Smart Nation—GWPrime*. Retrieved on 9 february 2022, from https://www.gwprime.geospatialworld.net/case-study/virtual-singapore-building-a-3d-empowered-smart-nation/

60. *Work with rules—ArcGIS CityEngine Resources | Documentation*. Retrieved on 22 february 2022, from https://doc.arcgis.com/en/cityengine/latest/help/help-working-with-rules.htm

61. Yadav, N. (2015). *An Introduction to Neural Network Methods for Differential Equations*. https://doi.org/10.1007/978-94-017-9816-7