



PhD Thesis SUMMARY

CONTRIBUTIONS TO THE SPATIAL REPRESENTATION OF BUILDINGS



PhD Supervisor: Professor PhD. Eng. Petre Iuliu Dragomir

PhD Candidate: Eng. Anca Patricia Grădinaru

**Bucharest
2023**

Table of Contents

1. Introduction	4
1.1. Motivation	4
1.2. Objectives and Purpose	4
1.3. The Structure of the PhD Thesis	4
2. State of the Art	6
2.1. 3D City Models and the „Smart City” Concept	6
2.2. Building Informational Model (BIM) and the “Digital Twin” Concept	6
2.3. Using Virtual Reality and Augmented Reality in Visualizing the 3D City Models	7
2.4. Open-source Geospatial Data	7
2.5. Conclusions	8
3. Geospatial Data Acquisition Methods	9
3.1. Geospatial Data Acquisition Using Laser Scanning	9
3.1.1. Aerial Laser Scanning	9
3.1.2. Terrestrial Laser Scanning	9
3.1.3. Mobile Laser Scanning	9
3.2. Geospatial Data Acquisition through Photogrammetry	10
3.2.1. The Fundamental Principle of Photogrammetry	10
3.2.2. Photogrammetric Images Classification	10
3.2.3. Photogrammetric Systems	10
3.3. Geospatial Data Acquisition for 3D City Model Generation	10
3.4. Conclusions	10
4. Spatial Modelling Techniques	12
4.1. LiDAR Point Cloud classification through Automatic Classification Methods	12
4.2. Utilizing Artificial Intelligence for LiDAR Point Cloud Classification	13
4.3. 3D Spatial Modelling Based on Photogrammetric Images	13
4.4. 3D Spatial Modelling Based on Street-View Imagery	13
4.5. Conclusions	14
5. Case Study	15
5.1. Point Cloud Classification	15
5.1.1. Study Area	15
5.1.2. Point Cloud Classification through automatic methods in ArcGIS Pro software	15
5.1.3. Point Cloud Classification using Bentley Microstation Terrascan Software Product	18
5.1.4. Extracting Data from Point Clouds Using Convolutional Neuronal Networks	19

5.1.5. Comparative Study and Conclusions	23
5.2. Buildings Footprint Extraction	26
5.2.1. Buildings Footprint Extraction Using the DL Algorithm from ArcGIS Pro	26
5.2.2. Training a DL Model in ArcGIS Pro for Building Footprint Extraction Using Photogrammetric Images	27
5.2.3. Buildings Footprint Extraction from the LiDAR Point Cloud	31
5.2.4. Comparative Study and Conclusions	32
5.3. Obtaining the 3D Buildings Model	35
5.3.1. 3D Modelling of the Buildings in ArcGIS Pro	35
5.3.2. 3D Buildings Modeling in CityEngine	37
5.3.3. Obtaining the 3D City Model Based on Open-Source Data	41
5.3.4. Visualizing 3D City Models	42
5.3.5. AR and VR Application for the Visualization of 3D City Models	42
5.4. Conclusions	45
6. General Conclusions, Original Contributions and Perspectives	47
6.1. General Conclusions	47
6.2. Original Contributions	50
6.3. Perspectives	51
References	52

1. Introduction

Today's society's problems require quick and effective solutions. The expansion of urban agglomerations is a well-known problem (Wei et al., 2023), and rapid population growth has a strong impact on the environment ("Population growth"). All this has led to the need to develop better methods of managing resources and emergency situations in urban areas. Such methods can be developed based on 3D models of cities, generated from quality geospatial data.

Recent research indicates that artificial intelligence simplifies the process of extracting geospatial data needed for 3D modelling, either based on point clouds or photogrammetric images. Also, virtual reality can be an effective framework for visualizing and managing the obtained models.

1.1. Motivation

Analyzing the current literature regarding this topic, I found that although there are many articles addressing the subject of 3D spatial modelling, they do not propose a complete workflow that analyzes the data from the perspective of the specific characteristics of the area and which also includes an efficient way of distributing and visualizing these models. Virtual reality and augmented reality are evolving technologies and can offer new perspectives in managing current problems that the society is facing.

1.2. Objectives and Purpose

The purpose of the doctoral thesis is to obtain the 3D models of buildings through an optimal workflow, and also to distribute and visualize these models, in order to increase the efficiency of their use. Thus, within the doctoral thesis, different geospatial data acquisition technologies will be analyzed at theoretical level and different 3D spatial modelling techniques will be tested, the study area being Baia Mare city.

The objectives of the PhD thesis are the analysis and comparison of methods for point cloud classification, the analysis and comparison of methods for extracting buildings footprints based on photogrammetric images, and the study of the possibilities of generating, distributing and visualizing 3D models of the buildings. After making this comparisons, an optimal workflow for the intended purpose will be proposed. Such a workflow can be further implemented by authorities or other interested persons.

1.3. The Structure of the PhD Thesis

The doctoral thesis is structured in six chapters.

In **Chapter 1 – Introduction** the importance of the subject, the motivation for the research, the objectives and purpose of the thesis and the structure of the thesis are presented.

Chapter 2 – State of the Art includes examples of 3D city models and information on standardisation of 3D models, CityGML standards in particular. Topics such as Smart City, BIM (Building Information Model) and Digital Twin are addressed and the importance of using virtual reality and augmented reality in visualizing 3D models of cities is highlighted. In addition, the possibility of using open-source geospatial data in 3D modelling is also analyzed.

Chapter 3 – Geospatial Data Acquisition Methods includes the analysis of geospatial data acquisition methods through airborne, terrestrial and mobile laser scanning and examples of laser scanning systems and workflow in laser scanning are presented. Moreover, the chapter includes information about methods of geospatial data acquisition through aerophotography, types of photogrammetric systems, examples of photogrammetric cameras and also the workflow in capturing photogrammetric images. At the end of the chapter, the instruments with which the geospatial data used in the case study of the PhD thesis were acquired are presented.

Chapter 4 – Spatial Modelling Techniques presents different methods for classifying LiDAR (Light Detection and Ranging) point clouds and extracting buildings footprints based on photogrammetric images, focusing on the use of convolutional neural networks. 3D spatial modelling based on Street-View images is also mentioned.

Chapter 5 – The Case Study covers point cloud classification, buildings footprints extraction, obtaining the 3D models and publishing and visualising 3D city models. For the point cloud classification, ArcGIS Pro and Bentley Microstation Terrascan software products were used and also a DL (Deep Learning) model trained for that area. At the end, the results were compared. The buildings footprints were extracted by three methods: using a DL model from ArcGIS Online, using a trained DL model specific to the study area, and from the LiDAR point cloud, and finally the results were compared. A comparison was also made with the footprint of a building exported from eTerra. 3D buildings models were generated in ArcGIS Pro and CityEngine. Modelling in CityEngine also involved advanced modelling of building facades, using the CGA (Computer Generated Architecture) programming language. A method for generating the city's 3D model based on open-source data was also presented. At the end of the chapter, methods for distributing and visualizing the 3D models obtained through ArcGIS Online and a visualization application in a VR (Virtual Reality) environment were presented and analyzed.

Chapter 6 – General Conclusions, Original Contributions and Perspectives contains the highlighting of original contributions found in the doctoral thesis, general conclusions on the results obtained and prospects for further research in the future.

2. State of the Art

2.1. 3D City Models and the „Smart City” Concept

A 3D model of the city is a three-dimensional geometric representation of urban objects and structures, especially buildings, which can be used in numerous applications, such as visibility analysis, energy demand estimation, shadow estimation, utility network management, energy potential estimation, 3D cadastre, infrastructure planning, sound propagation (Biljecki et al., 2015), emergency response (Grădinaru et al., 2022b), etc. A semantic 3D model of the city is a model that integrates the attributes of the objects, as well as different relationships between them. The semantic enrichment of 3D city models is a process that involves adding new information to an existing model to better connect it to the surrounding reality (Billen et al., 2014).

Standardising 3D city models enables data exchange and interoperability between them. CityGML standards, which define a conceptual model for representing 3D models of cities, are used to describe the complexity of building geometries (Biljecki et al., 2014) or the proximity between a 3D model of an object and the object itself (Tang et al., 2020).

Some examples of 3D city models can be the 3D BAG ("3D BAG Viewer"), a dataset containing 3D models of the buildings in the Netherlands, the 3D models of Helsinki city ("Helsinki 3D", 2023), or the Boston city model ("About 3D | Boston Planning & Development Agency").

The concept of "Smart City" has emerged as a solution for urban sustainability (Cai et al., 2023). Rapid population growth and migration from rural to urban areas have led to increased resource consumption, space requirements and pollution, and in this context, information and communication technologies integrated into an accessible infrastructure with available renewable resources, designed to limit consumption, can enable sustainability and embody the concept of "Smart City" (Liu et al., 2022). Geospatial data supports the smart city concept, providing information about location, neighborhoods, objects, networks, and accuracy (Guler and Yomralioglu, 2022).

2.2. Building Informational Model (BIM) and the "Digital Twin" Concept

The Building Informational Model is a digital representation of the physical and functional characteristics of a building (National Institute of Building Sciences buildingSMART alliance, 2015), being used in situations such as building structures monitoring (Xu et al., 2023a) or emergency evacuations planning (Xu et al., 2023b).

The concept of "Digital Twin" (DT) is a probabilistic simulation, which faithfully describes the state of the correspondent in reality, using both real-time data and historical data related to the building (Schrotter and Hürzeler, 2020). The difference

between BIM and DT is that while BIM provides static data, DT uses sensors and real-time data (Attaran and Celik, 2023).

2.3. Using Virtual Reality and Augmented Reality in Visualizing the 3D City Models

Cecotti (2022) defines virtual reality as a simulated experience. According to Li et al. (2022), virtual reality is an interface based on immersion, interaction and imagination, and the application of this technology in the urban environment can help manage 3D modelling of the city and virtual planning of certain projects.

Wang et al. (2018) propose a GIS analysis platform in VR, which serves for spatial analysis and visualization of 3D models and concluded that the 3D model visualization and analysis platform is a useful tool for both social service agencies and citizens, being used in exploring and analyzing city-related data directly.

While VR is a completely virtual environment and requires special glasses for visualization, augmented reality (AR) integrates the real world and virtual representations and can be accessed without special glasses, via a screen ("What's the Difference Between AR and VR?" 2020).

2.4. Open-source Geospatial Data

In a study conducted by Badea & Badea (2022), the usefulness of open geospatial data is emphasized, their characteristics are analyzed and a correlation with existing standards in the field is made. Initiatives such as OSM (Open Street Map), Wikimapia, data sources and tools such as those provided by NASA (National Aeronautics and Space Administration), Earth Observation, Copernicus Land Monitoring Service, Sentinel and also data provided by the INSPIRE directive were highlighted.

Taking into account the situation in Romania, several sources of geospatial data can be mentioned. ANCPI (National Agency for Cadastre and Real Estate) provides, through a geoportal, the situation of the buildings registered in the currently existing land book, as well as the existing buildings on the land in question ("Imobile eTerra - Public"). The TopRO50 plan, containing the vectorized buildings footprints, is provided by ANCPI and can be downloaded. Other data made available by ANCPI through the geoportal are: map sheets at 1:50 000 scale or data from the LAKI II project (Land Administration Knowledge Improvement). OSM (Open Street Map) is a digital representation of the entire world built by volunteers ("About OpenStreetMap - OpenStreetMap Wiki") and contains open data obtained from measurements, photogrammetric images or governmental data.

2.5. Conclusions

This chapter analyzed the current state of the art regarding 3D models of cities and the concept of "Smart City", the Building Informational Model (BIM) and the concept of "Digital Twin", the use of virtual reality and augmented reality in visualizing 3D models of cities and open-source geospatial data that can lay the groundwork for 3D spatial modelling.

3D models of cities have proven their utility in many areas, and CityGML standards help increase interoperability between different models. Thus, various standards have been defined, such as CityGML standards, which highlight five levels of detail. 3D city models standardisation allows data exchange between them, contributing to increasing the efficiency of the spatial modelling process.

The data acquisition technique and the level of detail of the models shall be chosen according to the field of use of the obtained model.

The „Smart City” concept has led to increased sustainability in urban environments and better resource management. Constantly changing urban areas require a virtual environment that allows the analysis of different scenarios in terms of resource consumption, space requirements or pollution levels. Such an environment also offers possibilities for planning and creating a comfortable and sustainable urban environment.

3D models of cities such as 3D BAG, Helsinki 3D, 3D model of Boston city or 3D Virtual Singapore model, which also adopts the concept of „Smart City”, were exemplified. Thus, it can be concluded that 3D models of cities can show their evolution over periods of time and can form the basis of various analyses related to energy, water consumption, heating, electricity or calculations of solar energy potential.

BIM and DT concepts use both real-time and historical building data, and bringing them into a GIS environment would help improve the entire 3D spatial modelling process.

Virtual reality and augmented reality support 3D modelling and contribute to improving the visualization of 3D models and their use, through virtual planning of certain projects. In addition, a virtual 3D model of the city is easier to visualize and understand by the general and non-specialist public. Therefore, I can conclude that virtual reality and augmented reality can significantly contribute to increasing the efficiency of the implementation of projects that involve multiple fields of applicability.

The basis for generating 3D models of cities can also be open-source geospatial data. Compliance with existing standards in the field is very important for this data to be used in an efficient way. Data such as those from OSM, data provided by NASA, Earth Observation, Copernicus Land Monitoring Service, Sentinel, data provided by the INSPIRE directive and also data made available by ANCPI were highlighted.

3. Geospatial Data Acquisition Methods

A first step towards obtaining 3D models is represented by the acquisition of geospatial data. Laser scanning and photogrammetry are the two main technologies for high-quality geospatial data acquisition (Figure 3.1).

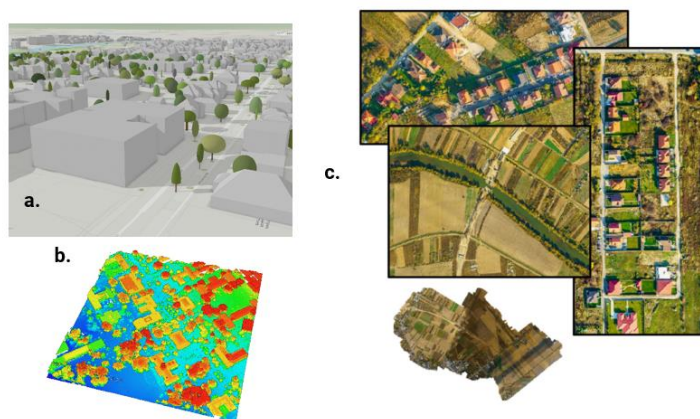


Figure 3.1. Examples of Geospatial Data Acquired for the Spatial Representation of Buildings (a.3D representation - capture taken in ArcGIS Pro program, data imported from ArcGIS Online Portal, b.LiDAR point cloud - capture taken in ArcGIS Pro software, c.images taken with a photogrammetric camera, within the „Cornel &Cornel Topoexim” company)

3.1. Geospatial Data Acquisition Using Laser Scanning

3.1.1. Aerial Laser Scanning



Figure 3.2. Point Cloud Obtained from Airborne Laser Scanning

3.1.2. Terrestrial Laser Scanning

3.1.3. Mobile Laser Scanning

3.2. Geospatial Data Acquisition through Photogrammetry

3.2.1. The Fundamental Principle of Photogrammetry

3.2.2. Photogrammetric Images Classification

3.2.3. Photogrammetric Systems

- Nadiral photogrammetric cameras;
- Oblique photogrammetric cameras.

3.3. Geospatial Data Acquisition for 3D City Model Generation

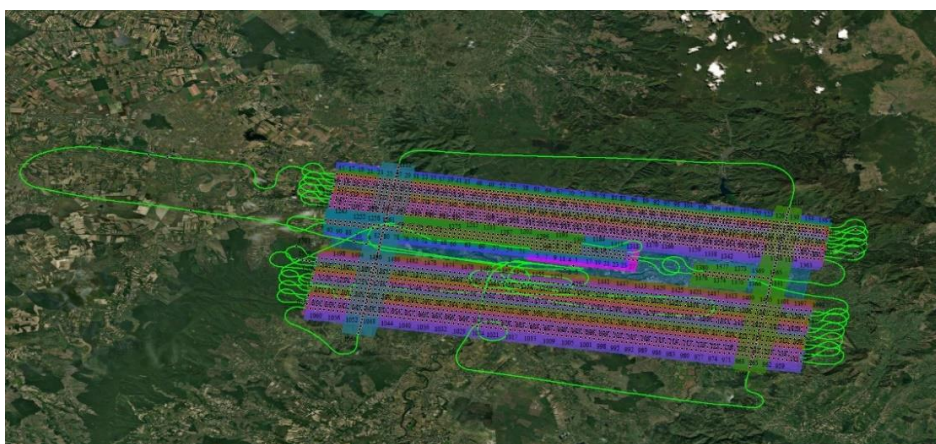


Figure 3.3. Flight Plan

The chosen study area is in Baia Mare, and the geospatial data were acquired during two flights (Figure 3.3), the first with a duration of 3h49m and the second with a duration of 2h45m.

For the acquisition of the geospatial data, a system consisting of a RIEGL VQ-780ii-S laser scanner, two Phase One photogrammetric cameras - P1-iXM-RS150F and P1-iXM-RS100F and a GNSS/INS AEROcontrol-II navigation system was used, mounted on board of a Tecnam P2006T drone.

3.4. Conclusions

A first step towards obtaining 3D models is represented by the acquisition of geospatial data. The quality and actuality of geospatial data is an important factor in accurate digital 3D modelling.

One method of geospatial data acquisition is laser scanning, which can be airborne, ground, or mobile. In this chapter I presented the main components of scanning systems and the operation mechanism of a laser scanner. In addition to the

scanning system, it is essential to have a reference GNSS station for differential positioning on the ground.

For airborne laser scanning, examples of modern scanning systems such as Leica CityMapper-2 and the RIEGL VQ-780ii-S Laser Scanner were highlighted.

The workflow in airborne laser scanning consists of study flight planning, study flight, and data processing. In terms of data processing, the automatic classification of the point cloud is the main challenge, due to the complex structures of the objects. The latest research in the field is based on machine learning algorithms such as SVM, Adaboost, Random Forest, Markov Random Field or CRF (Conditional Random Field). The latter proved to be more effective because it solves the problem of contextual information. In addition, recent research discusses the use of convolutional neural networks in the classification of point clouds, based on advanced DL models.

Terrestrial laser scanning allows us to obtain point clouds with a higher density than those obtained from airborne laser scanning. Various terrestrial laser scanners, such as Leica ScanStation P50 and Trimble TX8, as well as the workflow in terrestrial laser scanning were presented.

Mobile laser scanners integrate navigation technologies without the need for any other information, such as those provided by ground control stations. The possibilities of mobile laser scanning involve the "stop-and-go" observation method, in which scans are performed in a static mode, with the vehicle changing its position after each scan, and the "on-the-fly" observation method, in which the vehicle moves along a trajectory without stopping, and the laser scanner scans continuously. It can be concluded that the "on-the-fly" method is suitable for projects with a short execution time, as it has the advantage of a shorter scanning time, but each point collected is referred to an individual coordinate system, unlike the "stop-and-go" observation method, where each point cloud is defined in the scanner's local coordinate system.

Another geospatial data acquisition technique is aerophotography. This chapter presents classifications of photogrammetric images and different types of photogrammetric systems, which can be nadiral (e.g. Vexcel UltraCam Eagle Mark 3 or Leica DMC II) or oblique (e.g. Vexcel UltraCam Osprey, Leica RCD30 Oblique, IGI Quattro DigiCAM Oblique). The main steps in taking photogrammetric images were presented and the importance of ground control points, which need to be easy to identify and determine by GNSS measurements, was emphasized. The characteristics of digital images play an important role in obtaining quality geospatial data based on orthorectified images. It can be concluded that especially a high spatial resolution of the image leads to obtaining geospatial data with high accuracy.

4. Spatial Modelling Techniques

4.1. LiDAR Point Cloud classification through Automatic Classification Methods

To obtain a 3D city model based on LiDAR data, the point cloud needs to be classified accordingly. There are numerous software products that enable this. One of these is ArcGIS Pro, which uses the classification model defined by the American Society for Photogrammetry and Remote Sensing (ASPRS), which is structured according to Table 4.1 ("Change LAS Class Codes (3D Analyst)–ArcGIS Pro | Documentation").

Table 4 1. Point Classes Used in ArcGIS Pro

Classification Code	Class
0	Never classified
1	Unassigned
2	Ground
3	Low Vegetation
4	Medium Vegetation
5	High Vegetation
6	Building
7	Low Point
8	Key model / Reserved
9	Water
10	Rail
11	Road Surface
12	Overlaps / Reserved
13	Wire – Guard
14	Wire – Conductor
15	Transmission Tower
16	Wire – Connector
17	Bridge Deck
18	High Noise
19-63	Reserved for ASPRS definitions
32-255	Reserved for user definitions

4.2. Utilizing Artificial Intelligence for LiDAR Point Cloud Classification

Artificial Intelligence (AI) is a technology that uses a computer to simulate human intelligence and trains the computer to further learn human behaviors for decision-making (Zhang & Lu, 2021).

Neural networks are a concept of artificial intelligence, inspired by the human brain, that replicate how human neurons transmit signals to each other. They are composed of nodes layers: an input layer, one or more hidden layers, and an output layer. Convolutional neural networks differ from other neural networks by their superiority in image, speech or audio input data and by the three main layers: the convolutional layer, the grouping layer and the fully connected layer. There are different types of convolutional neural network architectures, such as AlexNET, VGGNet, GoogLeNet, ResNet, ZFNet ("What are Convolutional Neural Networks?").

4.3. 3D Spatial Modelling Based on Photogrammetric Images

The image classification process involves applying specific rules, designed based on spectral or textural characteristics of the image, to assign labels to groups of pixels or vectors in that image. Classification can be supervised, by manually selecting training data and assigning it to the appropriate class, or unsupervised, which uses specific algorithms for fully automatic image classification (Jog and Dixit, 2016).

The generation of buildings footprints can be achieved by training a DL model based on a MaskRCNN architecture ("Automated Building Footprint Extraction using Deep learning"), developed from the need to improve techniques for detecting objects in images. In order to obtain the most regular forms of buildings footprints, there is the possibility of geoprocessing that allows the use of a polyline compression algorithm to correct their distortions ("Regularize Building Footprint (3D Analyst)–ArcGIS Pro | Documentation").

4.4. 3D Spatial Modelling Based on Street-View Imagery

Another source of data for 3D spatial modelling of cities can be street-view images. Biljecki & Ito, 2021 conducted a study on the use of street-view imagery in urban analysis and GIS and concluded that most existing models rely mainly on Google Street View, but other companies' focus on street-view products can open up new horizons and improve data, including greater coverage and images of building interiors.

4.5. Conclusions

In this chapter, different spatial modelling techniques have been analyzed. Data processing involves classifying the LiDAR point cloud and extracting geometric elements from photogrammetric images, in order to obtain the 3D models of the buildings.

Recent research has led to the demonstration of the effectiveness of AI involvement in LiDAR point cloud classification. Using a PointCNN convolutional neural network involves preparing a training data file and a validation data file. The more data these files present, the greater the accuracy of the trained classification model.

Buildings footprints can also be extracted from photogrammetric images. Image classification can be supervised, by manually selecting data and assigning it to the appropriate class, or unsupervised, which uses specific algorithms for fully automatic image classification. Training a DL model can be based on a MaskRCNN architecture, used to detect objects in the image. Therefore, the training data should be a file with the previously determined buildings footprints. By training such a model for a specific area, the buildings footprints can be obtained with higher accuracy than using an existing model, trained for another type of area. The more data the training data file contains, the higher the accuracy of the model obtained.

Another data source for 3D spatial modelling of cities can be street-view images, by training specific DL models, but, due to the nature of these images, the accuracy is lower than the models obtained through the methods presented above.

In conclusion, I proposed the workflow in Figure 4.1 to obtain the 3D model of a city, and in the case study different working methods will be analyzed and compared.

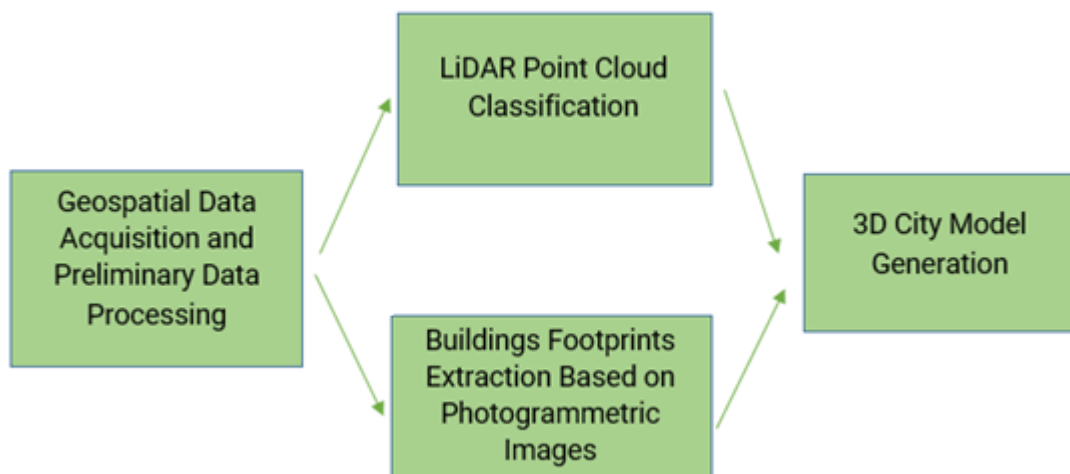


Figure 4.1. Workflow for Obtaining 3D City Models

5. Case Study

5.1. Point Cloud Classification

5.1.1. Study Area

The chosen study area is in Baia Mare (Figure 5.1).



Figure 5.1. Study Area

5.1.2. Point Cloud Classification through automatic methods in ArcGIS Pro software

The file in *.LAS (LASer) format contains a total of 44,913,903 points. The points belonging to the classes *Ground*, *Building* and *Low*, *Medium* and *High Vegetation* were classified. The classification results can be found in Figures 5.2, 5.3 and 5.4.

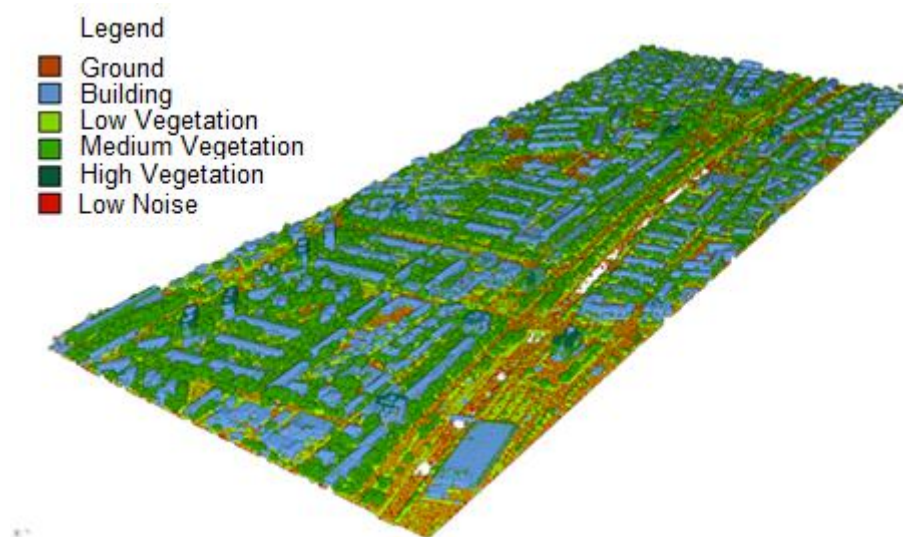


Figure 5.2. Classified Point Cloud

Item	Category	Pt_Cnt	Percent	Z_Min	Z_Max
1_Unclassified	ClassCodes	32197	0.07	205.83	248.08
2_Ground	ClassCodes	7735046	17.22	201.08	222.9
3_Low_Vegetation	ClassCodes	11271865	25.1	201.17	227.26
4_Medium_Vegetation	ClassCodes	9548292	21.26	207.38	243.22
5_High_Vegetation	ClassCodes	81217	0.18	232.09	264.41
6_Building	ClassCodes	10161344	22.62	202.55	258.91
7_Low_Point(noise)	ClassCodes	6083682	13.55	193.85	222.66
14_Reserved	ClassCodes	43	0	216.73	222.42
18_Reserved	ClassCodes	217	0	202.03	274.55

Figure 5.3. Point Cloud Classification Results with ArcGIS Pro Software

Point Cloud Classified with ArcGIS Pro

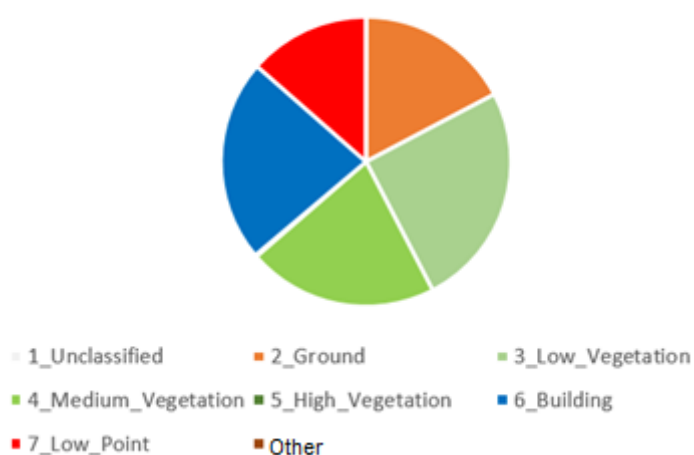


Figure 5.4. Point Cloud Classification Results Graph

By using this method, some points have been misclassified. The main error in classification was the classification of some building elements as vegetation points (Figure 5.5). These errors require manual intervention.

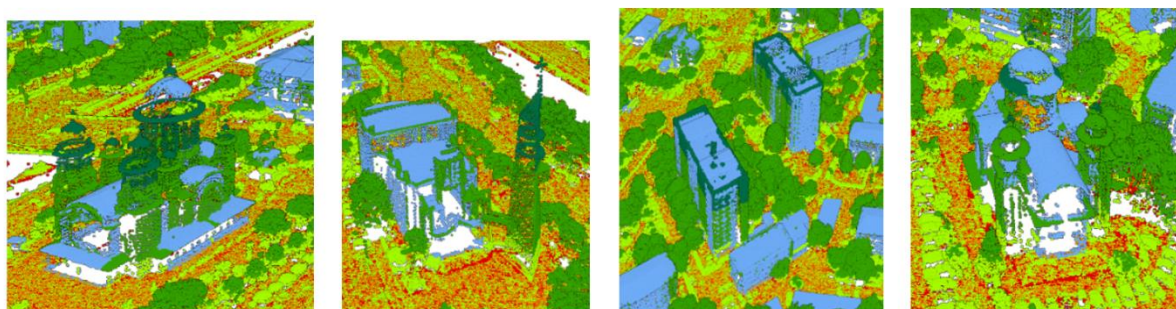


Figure 5.5. Classification Errors with ArcGIS Pro Software

For a larger volume of data, I proposed using ArcGIS Pro ModelBuilder. I created the model in Figure 5.6., which follows the point cloud classification steps outlined earlier ("Use ModelBuilder—ArcGIS Pro | Documentation").

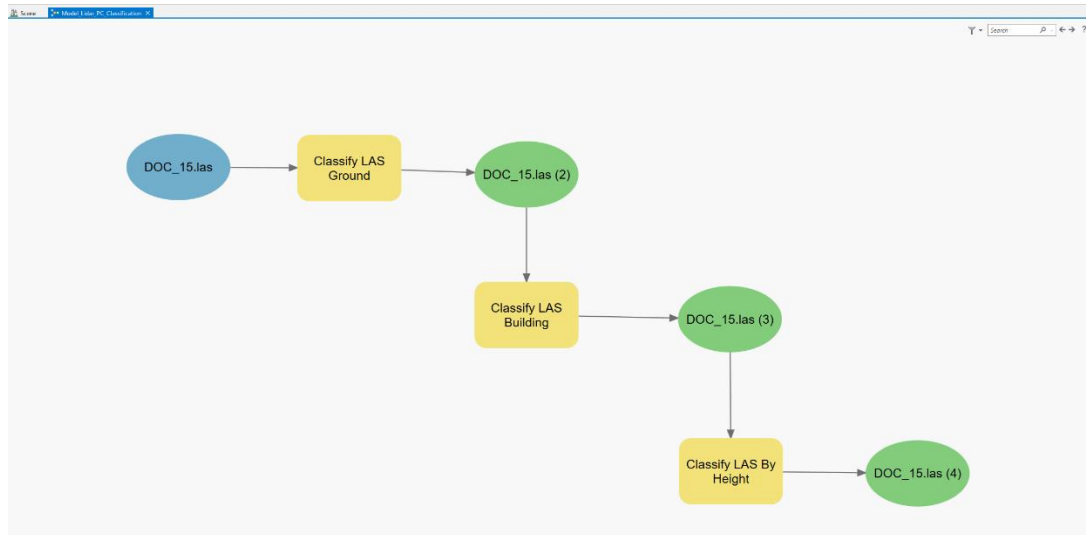


Figure 5.6. Point Cloud Classification Model in ArcGIS Pro ModelBuilder

Another way to make the point cloud classification process more automatic in ArcGIS Pro is to use ArcGIS API (Application Programming Interface) for Python, a programming language used to visualize and analyze data in a GIS environment ("Overview of the ArcGIS API for Python"), by using the following code sequence:

```
import arcpy
```

```
def Model_Lidar_PC_Classification(): # Model_Lidar_PC_Classification
```

```
    # To allow overwriting outputs change overwriteOutput option to True.
```

```
    arcpy.env.overwriteOutput = False
```

```
    # Check out any necessary licenses.
```

```
    arcpy.CheckOutExtension("3D")
```

```
    DOC_15_las = "DOC_15.las"
```

```
    # Process: Classify LAS Ground (Classify LAS Ground) (3d)
```

```
    DOC_15_las_2_ = arcpy.ddd.ClassifyLasGround(in_las_dataset=DOC_15_las,
method="CONSERVATIVE", reuse_ground="RECLASSIFY_GROUND", dem_resolution="",
compute_stats="COMPUTE_STATS", extent="23.5528862297105 47.6563134028331
23.5716906627965 47.6633591935169", boundary="", process_entire_files="PROCESS_EXTENT",
update_pyramid="UPDATE_PYRAMID")[0]
```

```
    # Process: Classify LAS Building (Classify LAS Building) (3d)
```

```
    DOC_15_las_3_ = arcpy.ddd.ClassifyLasBuilding(in_las_dataset=DOC_15_las_2_, min_height="2
Meters", min_area="6 SquareMeters", compute_stats="COMPUTE_STATS",
extent="23.5528862297105 47.6563134028331 23.5716906627965 47.6633591935169",
boundary="", process_entire_files="PROCESS_EXTENT", point_spacing="",
reuse_building="RECLASSIFY_BUILDING",
photogrammetric_data="NOT_PHOTOGAMMETRIC_DATA", method="STANDARD",
classify_above_roof="NO_CLASSIFY_ABOVE_ROOF", above_roof_height="1.5 Meters",
above_roof_code=6, classify_below_roof="CLASSIFY_BELOW_ROOF", below_roof_code=6,
update_pyramid="UPDATE_PYRAMID")[0]
```

```
# Process: Classify LAS By Height (Classify LAS By Height) (3d)
DOC_15_las_4 = arcpy.ddd.ClassifyLasByHeight(in_las_dataset=DOC_15_las_3,
ground_source="GROUND", height_classification=[[3, 5], [4, 25], [5, 50]], noise="ALL_NOISE",
compute_stats="COMPUTE_STATS", extent="23.5528862297105 47.6563134028331
23.5716906627965 47.6633591935169", process_entire_files="PROCESS_EXTENT", boundary="",
update_pyramid="UPDATE_PYRAMID")[0]

if __name__ == '__main__':
    # Global Environment settings
    with
    arcpy.EnvManager(scratchWorkspace=r"D:\Doctorat\3_An_III\5_Teza\4_1_Clasificarea_norului_de_p
uncte\ArcGIS\V2\V2\model_builder\MyProject74\MyProject74.gdb",
workspace=r"D:\Doctorat\3_An_III\5_Teza\4_1_Clasificarea_norului_de_puncte\ArcGIS\V2\V2\model
_builder\MyProject74\MyProject74.gdb"):
        Model_Lidar_PC_Classification()
```

5.1.3. Point Cloud Classification using Bentley Microstation Terrascan Software Product

With the Bentley Microstation Terrascan software product, points belonging to the classes *Ground*, *Building*, *Low Vegetation*, *Medium Vegetation* and *High Vegetation* were classified. The classification results can be found in Figures 5.7, 5.8 and 5.9.

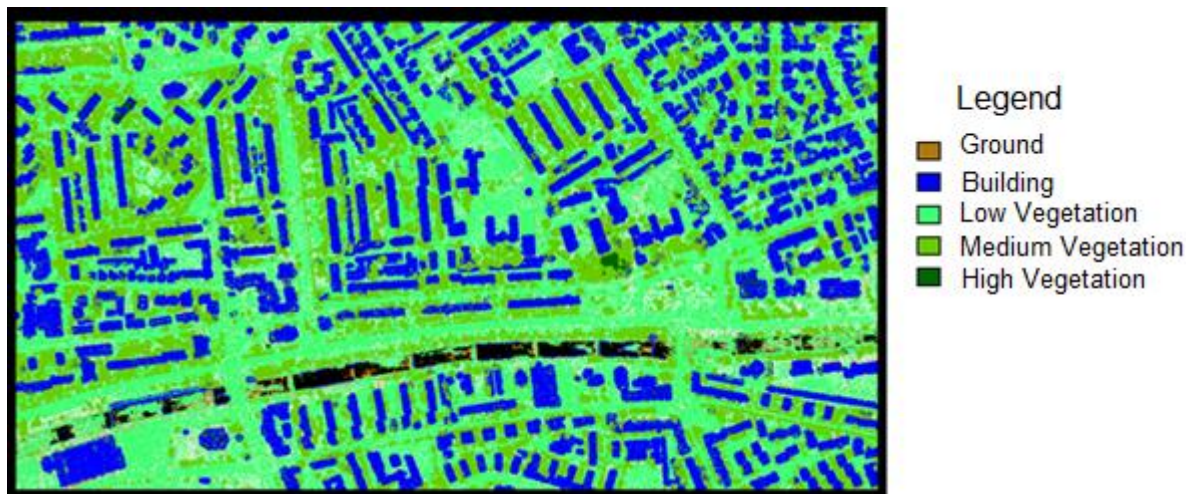


Figure 5.7. Point Cloud Classified with Bentley Microstation Terrascan Software

All points		44 913 903	193.85	274.55
Active points		44 913 903		
Neighbour points		0		
Class	Description	Count	Min Z	Max Z
1	Default	4 877 731	201.15	261.70
2	Ground	8 537 934	201.02	222.73
3	Low vegetation	11 773 888	201.16	227.29
4	Medium vegetation	10 034 580	207.38	243.07
5	High vegetation	96 482	231.36	264.41
6	Building	9 591 490	201.61	260.12
7	Low point	1 556	193.85	222.13
8	Model keypoints	0	-	-
14	Class 14	43	216.73	222.42
18	Class 18	199	202.03	274.55

Figure 5.8. Point Cloud Classification Results with Bentley Microstation Terrascan Software

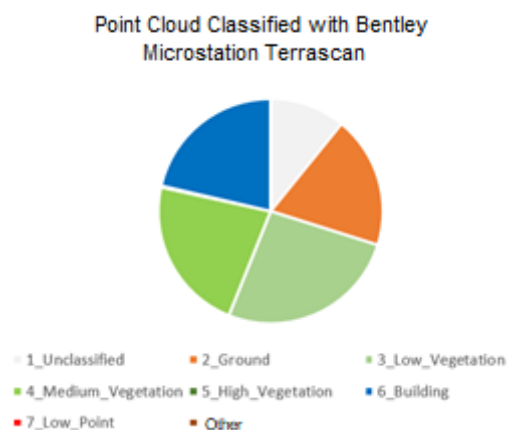


Figure 5.9. Point Cloud Classification Results Graph

As far as classification errors are concerned, the most common error was the classification of walls as vegetation elements. Also, another common error was the classification of ground or low vegetation elements in the proximity of buildings as elements of the buildings. These errors can be seen in Figure 5.10 and require manual intervention.

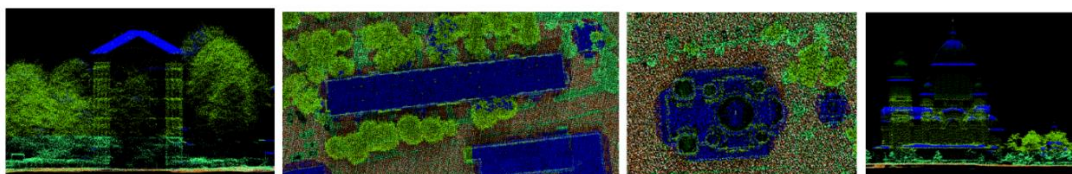


Figure 5.10. Building Walls Classified as Vegetation Elements

5.1.4. Extracting Data from Point Clouds Using Convolutional Neuronal Networks

Since convolutional neural networks are data-driven models, the effectiveness of these models in their application domains is given by how well the training data

represent the area (Kattenborn et al., 2021). Next, the point cloud will be classified with DL algorithms and convolutional neural networks. When training a PointCNN model, the point cloud is divided into blocks of points containing a certain number of elements ("Point cloud classification using PointCNN").

Two file types are required to train a PointCNN model: a training data file and a validation data file. These files were created using ArcGIS Pro's existing automatic classification methods, with the classification errors manually corrected. In the end, I obtained the two datasets (Figure 5.11), with a total of 68% training data and 32% validation data.

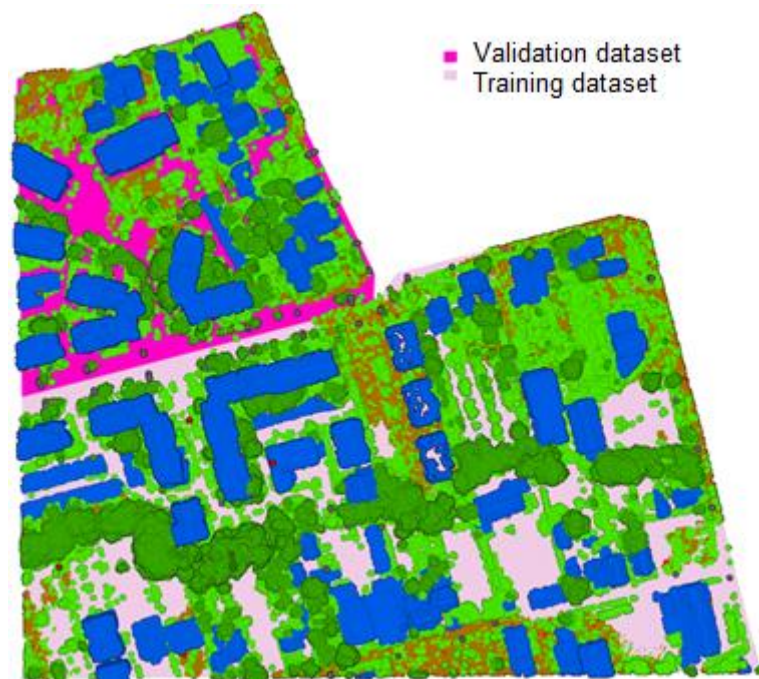


Figure 5.11. Training and Validation Datasets

The training and validation data have been prepared and the result can be found in Figures 5.12 and 5.13.

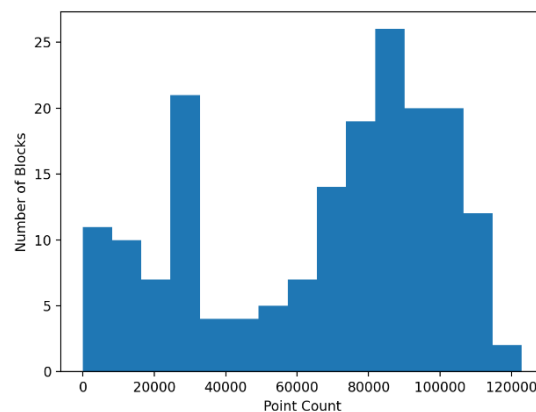


Figure 5.12. Histogram of Point Blocks in Training Data

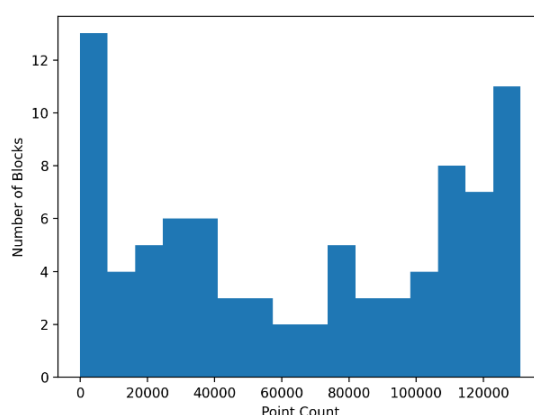


Figure 5.13. Histogram of Point Blocks in Validation Data

To train the model, I used the Python programming language in ArcGIS Pro and chose 20 training epochs:

```
from arcgis.learn import export_point_dataset, prepare_data, PointCNN
output_path=r'D:\Doctorat\3_An_III\5_Teza\4_1_Clasificarea_norului_de_puncte\Deep_learning_ArcGIS\V4\Model_train\Model_training_2\export2.pctd'
data = prepare_data(output_path, dataset_type='PointCloud', batch_size=2)
pointcnn = PointCNN(data)
pointcnn.fit(20)
```

The results of model training from each epoch can be seen in Figure 5.14 and are as it follows:

epoch	train_loss	valid_loss	accuracy	precision	recall	f1	time
0	2.017702	1.939532	0.420038	0.166875	0.162660	0.141807	15:11
1	1.719348	1.835655	0.478760	0.238613	0.218548	0.173044	14:07
2	1.416687	2.675217	0.428926	0.315331	0.270709	0.202408	13:59
3	1.066752	6.231255	0.305902	0.283549	0.239897	0.155649	13:58
4	0.927440	1.815611	0.484227	0.369048	0.326948	0.266849	13:58
5	0.834291	1.367668	0.560290	0.418993	0.370895	0.317546	13:58
6	0.800354	0.895765	0.669276	0.483309	0.407822	0.387601	13:56
7	0.750165	0.918152	0.672131	0.482960	0.421462	0.395064	13:57
8	0.731268	0.801657	0.690178	0.510082	0.410125	0.402040	13:57
9	0.716202	0.735054	0.708160	0.528342	0.408976	0.406577	14:02
10	0.682229	0.697355	0.720835	0.539543	0.431926	0.435531	14:24
11	0.669484	0.692361	0.726409	0.540671	0.437687	0.441238	14:08
12	0.665540	0.699732	0.733367	0.561855	0.458471	0.464235	13:55
13	0.656838	0.720050	0.721775	0.553432	0.428550	0.431470	13:56
14	0.641315	0.680481	0.734322	0.566254	0.447624	0.459272	13:55
15	0.629172	0.688940	0.728866	0.562753	0.439414	0.450087	14:10

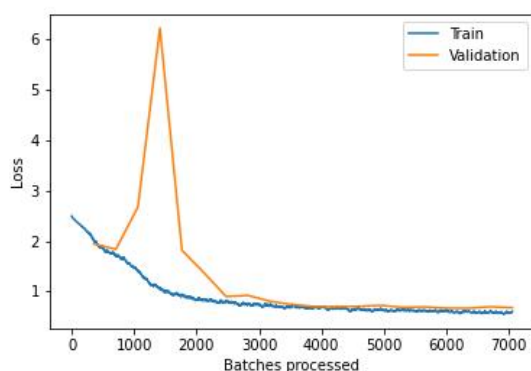
16	0.631952	0.664263	0.733555	0.574948	0.446280	0.458066	13:58
17	0.612834	0.662378	0.736640	0.574113	0.451032	0.462522	13:48
18	0.610737	0.691800	0.730252	0.569812	0.447568	0.456362	13:55
19	0.603156	0.673858	0.732018	0.573324	0.442920	0.451286	13:48

PointCNN

Backbone: None

Learning Rate: 4.7863e-04

Training and Validation loss



Sample Results

Ground Truth / Predictions (Displaying randomly sampled 20000 points.)

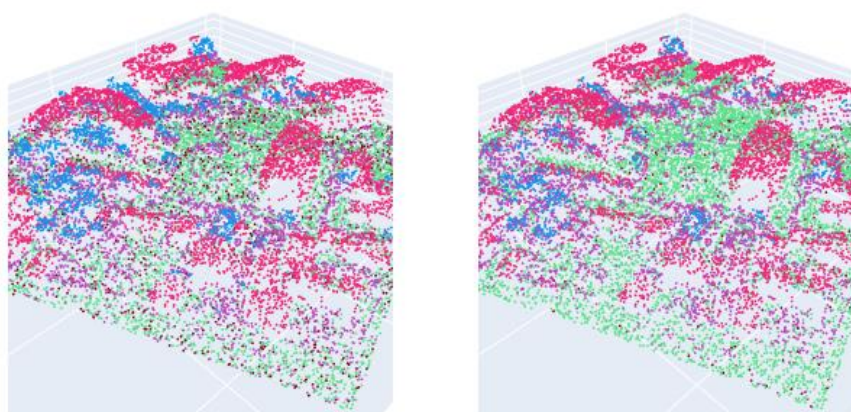


Figure 5.14. DL Model Training Results

Next, I classified the point cloud with the trained model and obtained the results from Figures 5.15, 5.16 and 5.17.

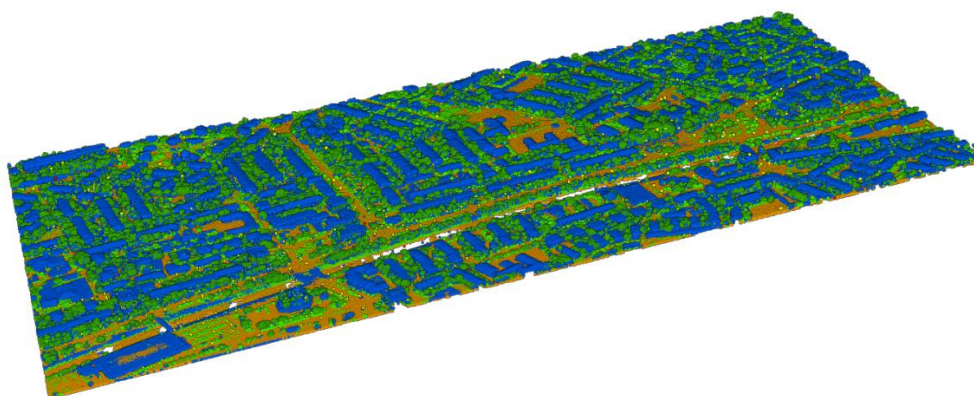


Figura 5.15. Point Cloud Classification Results with the DL Trained Model

Field: Add Calculate		Selection: Select By Attributes Zoom To Switch				
	Item	Category	Pt_Cnt	Percent	Z_Min	Z_Max
12	All	Returns	44913903	100	193.85	274.55
13	2_Ground	ClassCodes	18335153	40.82	201.1	243.51
14	3_Low_Vegetation	ClassCodes	6468741	14.4	201.25	254.58
15	4_Medium_Vegetation	ClassCodes	6114374	13.61	205.5	264.25
16	6_Building	ClassCodes	13696156	30.49	201.71	264.41
17	7_Low_Point(noise)	ClassCodes	220791	0.49	193.85	226.4
18	18_Reserved	ClassCodes	165	0	201.91	274.55
19	19_Reserved	ClassCodes	78523	0.17	201.94	272.85
20	Return_No	Attributes	<Null>	<Null>	<Null>	<Null>

Figure 5.16. Point Cloud Classification Results with the DL Trained Model

Point Cloud Classified with
the DL Trained Model

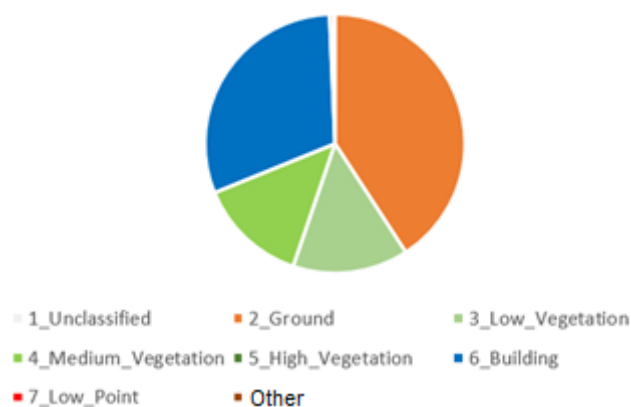


Figure 5.17. Point Cloud Classification Results Graph

5.1.5. Comparative Study and Conclusions

After classifying the point cloud using the three methods presented above, I obtained the results from Table 5.1 and Figure 5.18.

Table 5-1. Point Cloud Classification Results

Class/Method		ArcGIS Pro	Bentley Microstation TerraScan	ArcGIS Pro – Trained DL Model
No. of points	Ground	7,735,046	8,537,934	18,335,153
	Low Vegetation	11,271,865	11,773,888	6,468,741
	Medium Vegetation	9,548,292	10,034,580	6,114,374
	High Vegetation	81,217	96,482	-
	Building	10,161,344	9,591,490	13,696,156
	Low Noise	6,083,682	1,556	220,791
	Unclassified	-	4,877,731	-

From Table 5.1 it can be seen that by classifying the point cloud with ArcGIS Pro and Bentley Microstation Terrascan software, I obtained similar results. This is also due to the fact that similar classification parameters have been chosen for the two methods. One factor that determined the difference between the numbers of points assigned to the Building class is that the Bentley Microstation Terra Scan software did not classify the walls of the buildings, but only their roofs. In Figure 5.19, it can be seen that the points belonging to the walls of the building were classified as medium vegetation points, when classified with the Bentley Microstation Terrascan software. There is also a considerable difference in Ground Points due to the fact that when classified with ArcGIS Pro software, many of them were classified as noise points, shown in red in Figure 5.19-a, and when classified with Bentley Microstation Terrascan software, they remained unclassified, shown in white in Figure 5.19-b.

Comparative Graph of the Point Cloud Classification Methods

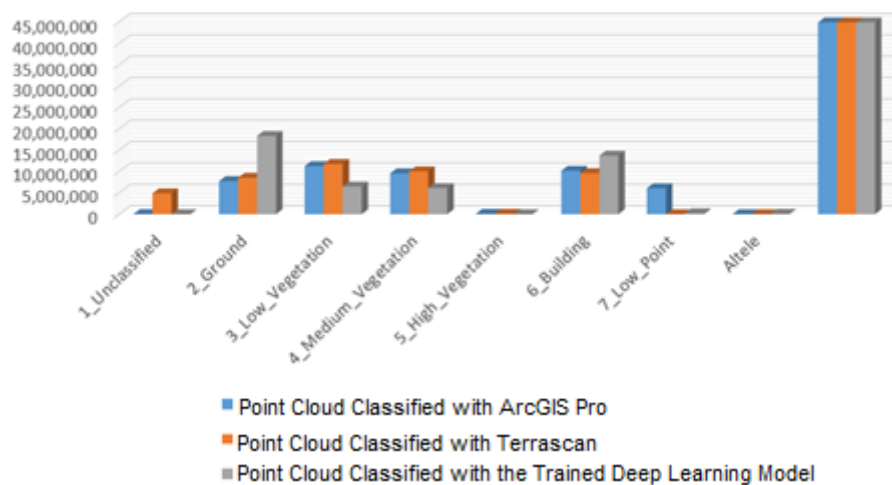


Figure 5.18. Comparative Graph of the Point Cloud Classification Methods

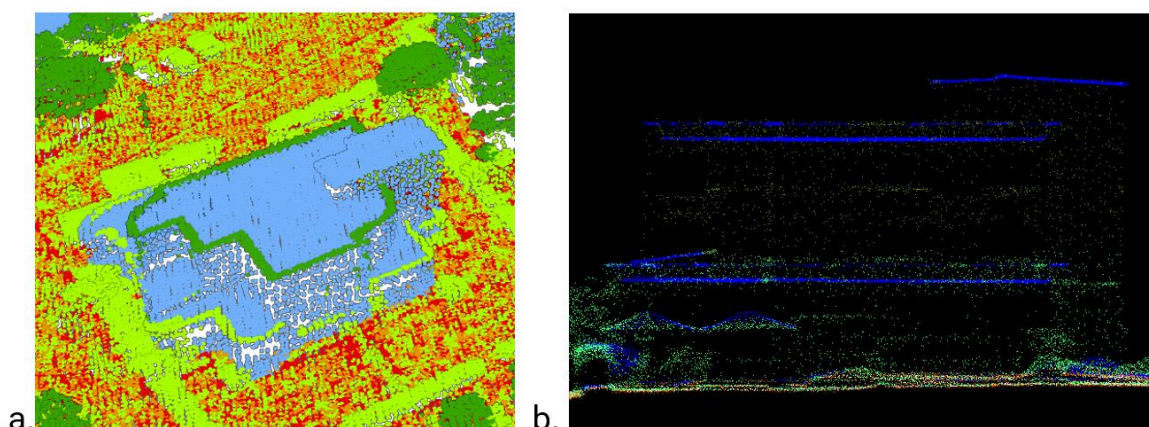


Figure 5.19. Point Cloud Classified in ArcGIS Pro (a.) and Point Cloud Classified with Bentley Microstation Terrascan (b.) – same building

Regarding the classification of the point cloud with the trained DL model, greater differences are found compared to the first two methods. Looking at the classified point cloud (Figure 5.20), it can be seen that many points of medium vegetation have been classified as building points, and also many points that make up flat surfaces have been interpreted as building roofs, although they are at ground level (Figure 5.21).

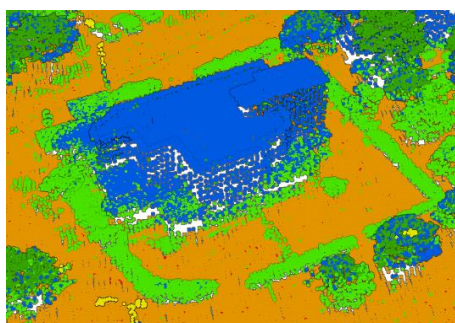


Figure 5.20. Point Cloud Classification with the Trained DL Model – same building found in Figure 5.19

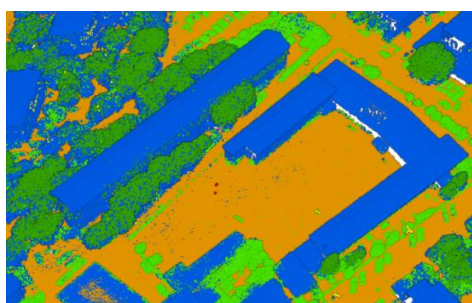


Figure 5.21. Point Cloud Classification with the Trained DL Model

For more complex buildings (Figure 5.22), all three methods proved insufficient to use a fully automatic classification, with the point cloud requiring manual intervention each time. The main problem lies in the fact that the irregular shapes of the roofs make many of their points to be assigned different vegetation classes.

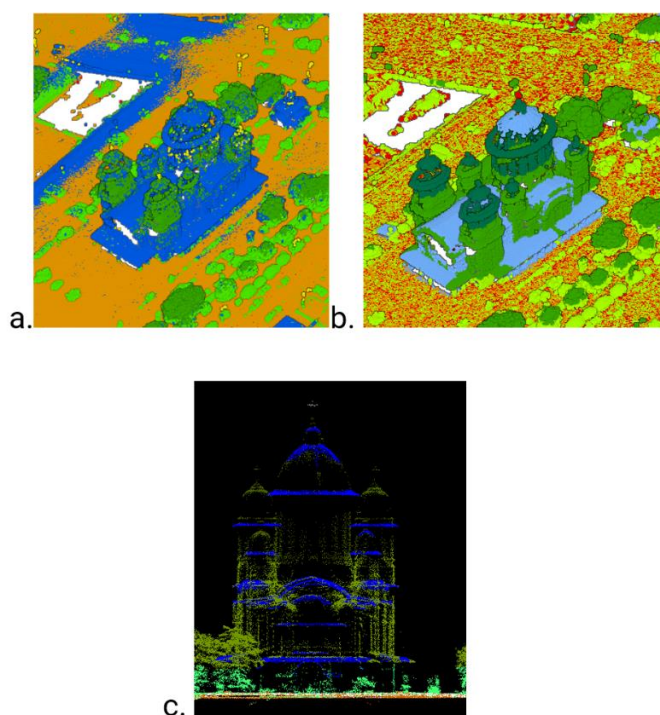


Figure 5.22. Building Classified Using the DL Trained Model (a.), ArcGIS Pro (b.) and Bentley Microstation Terrascan Software (c.)

In conclusion, the three point cloud classification methods resulted in similar results, but the most effective in terms of results and manual error correction interventions was the use of ArcGIS Pro's automated point cloud classification tools, which can be more efficient for classifying a larger data volume, such as that utilized for the case study in the doctoral thesis. In addition to a smaller number of classification errors, this method also allows you to make the classification process more automatic, by using the ArcGIS API for Python.

5.2. Buildings Footprint Extraction

5.2.1. Buildings Footprint Extraction Using the DL Algorithm from ArcGIS Pro

In ArcGIS Living Atlas there are different models for extracting buildings footprints based on satellite imagery (Gradinaru, 2022). I chose from Living Atlas the Building Footprint Extraction – USA model, based on a MaskRCNN architecture implemented using ArcGIS API for Python ("Building Footprint Extraction - USA - Overview"). The DL model was run on an input raster file to produce a feature class that will contain the found objects ("Detect Objects Using Deep Learning (Image Analyst)—ArcGIS Pro | Documentation"). The model is trained to identify the roofs of the buildings, which will be represented in the form of polygons. In total, 464 geometries were detected. To correct the polygons' distortions, I used a polyline

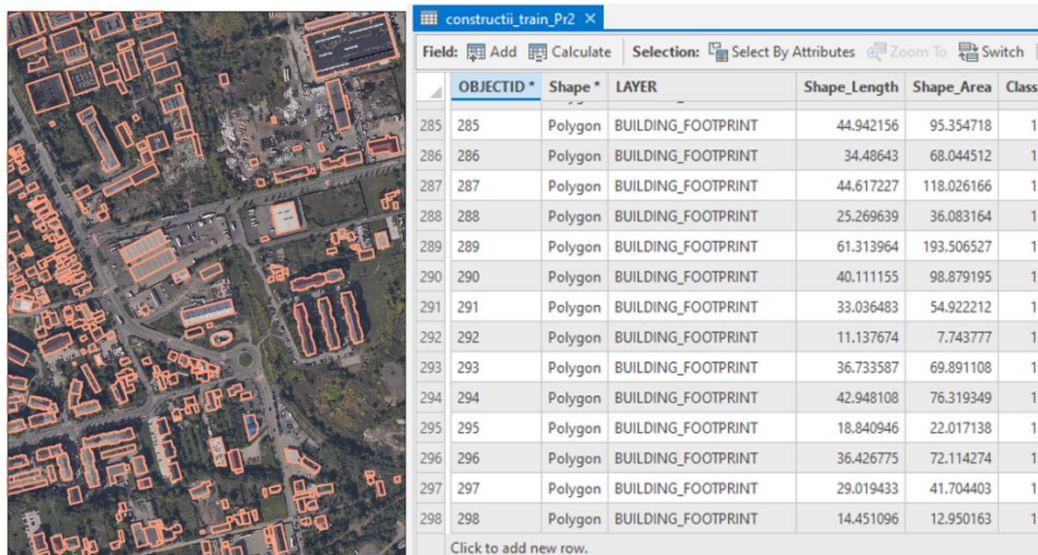
compression algorithm ("Regularize Building Footprint (3D Analyst)—ArcGIS Pro | Documentation"). Then I removed the overlaps and got a total of 390 geometries. The results can be found in Figure 5.23.



Figure 5.23. The Results of Extracting Buildings Footprints with the DL Model in ArcGIS Living Atlas

5.2.2. Training a DL Model in ArcGIS Pro for Building Footprint Extraction Using Photogrammetric Images

For training the Deep Learning model, I prepared a training dataset consisting of manually vectorized geometries for 298 buildings (Figure 5.24), which were assigned class 1 in the corresponding attribute table.



	OBJECTID *	Shape *	LAYER	Shape_Length	Shape_Area	Class
285	285	Polygon	BUILDING_FOOTPRINT	44.942156	95.354718	1
286	286	Polygon	BUILDING_FOOTPRINT	34.48643	68.044512	1
287	287	Polygon	BUILDING_FOOTPRINT	44.617227	118.026166	1
288	288	Polygon	BUILDING_FOOTPRINT	25.269639	36.083164	1
289	289	Polygon	BUILDING_FOOTPRINT	61.313964	193.506527	1
290	290	Polygon	BUILDING_FOOTPRINT	40.111155	98.879195	1
291	291	Polygon	BUILDING_FOOTPRINT	33.036483	54.922212	1
292	292	Polygon	BUILDING_FOOTPRINT	11.137674	7.743777	1
293	293	Polygon	BUILDING_FOOTPRINT	36.733587	69.891108	1
294	294	Polygon	BUILDING_FOOTPRINT	42.948108	76.319349	1
295	295	Polygon	BUILDING_FOOTPRINT	18.840946	22.017138	1
296	296	Polygon	BUILDING_FOOTPRINT	36.426775	72.114274	1
297	297	Polygon	BUILDING_FOOTPRINT	29.019433	41.704403	1
298	298	Polygon	BUILDING_FOOTPRINT	14.451096	12.950163	1

Figure 5.24. Training Data

The first step in training the model was to export the training data in RCNN Masks format ("Export Training Data for Deep Learning (Image Analyst)—ArcGIS Pro | Documentation"). Next, I converted the vector data into training datasets based on the

raster image. The result was a set of image chips measuring 256x256 pixels and a set of metadata files in the specified format. The next step was to train the Deep Learning model with the data obtained in the previous step. The results of the training during the 20 epochs were the following:

Start Time: Sunday, April 30, 2023 23:07:32

Learning Rate - slice(4.365158322401661e-06, 4.365158322401661e-05, None)

epoch	training loss	validation loss
0	1.1618988513946533	1.2790905237197876
1	1.1265311241149902	0.9537186622619629
2	1.067383885383606	0.8906569480895996
3	1.1501880884170532	0.8541356921195984
4	1.0023037195205688	0.7756800055503845
5	0.9016749858856201	0.7570561170578003
6	0.7635862231254578	0.731697678565979
7	0.8380852341651917	0.6944260597229004
8	0.7321746349334717	0.6711621284484863
9	0.9564940929412842	0.6712456345558167
10	0.7845675945281982	0.6268543004989624
11	0.656985342502594	0.5794755816459656
12	0.8817278146743774	0.607085108757019
13	0.582132875919342	0.5567061305046082
14	0.6240612864494324	0.5660204887390137
15	0.5917166471481323	0.5256257057189941
16	0.5708335638046265	0.5232657194137573
17	0.5908197164535522	0.5226365923881531
18	0.5473408102989197	0.5186260342597961
19	0.5487155914306641	0.5161070823669434

{'average_precision_score': {'1': 0.8224424088745176}}

Succeeded at Sunday, April 30, 2023 23:32:53 (Elapsed Time: 25 minutes 20 seconds)

Next, I used the trained model to obtain the footprints of the buildings, and then corrected the distortions of the polygons. I obtained a total of 1262 polygons. The last step was to remove overlapping areas and to merge overlapping geometries into a single geometry. I obtained a total of 628 polygons and the result can be found in Figure 5.25.



Figure 5.25. Buildings Footprints Generated with the Trained DL Model

For managing a larger number of photograms, I proposed using ArcGIS Pro ModelBuilder (Figure 5.26).

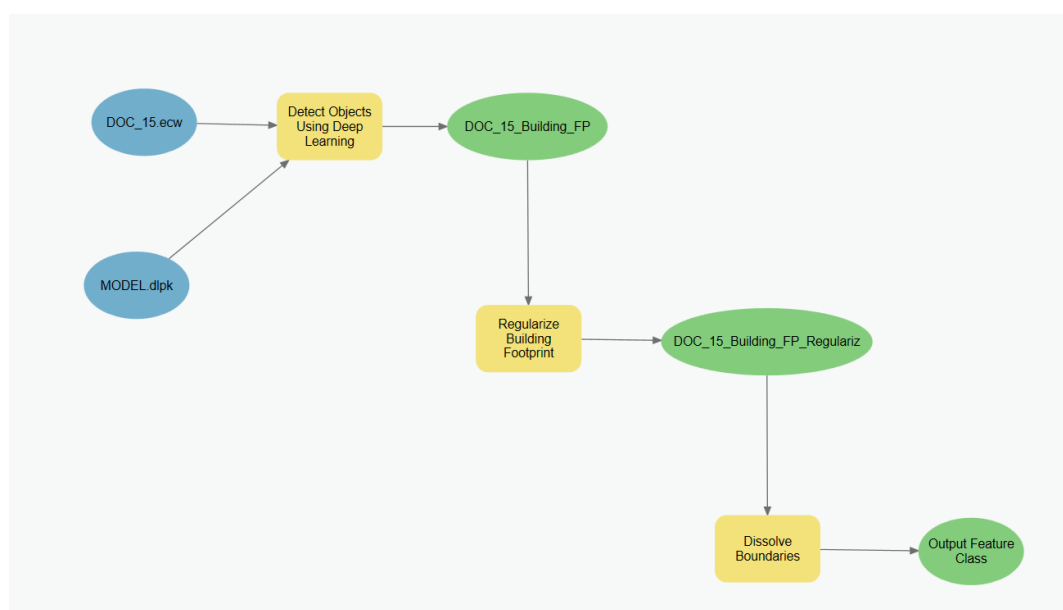


Figure 5.26. Buildings Footprints Extraction Model in ArcGIS Pro Model Builder

ArcGIS API for Python can be used to make the buildings footprints extraction process automatic, with the following code sequence:

```

import arcpy

def Model(): # Model

    # To allow overwriting outputs change overwriteOutput option to True.
    arcpy.env.overwriteOutput = False

    # Check out any necessary licenses.
    
```

```

arcpy.CheckOutExtension("ImageExt")
arcpy.CheckOutExtension("ImageAnalyst")
arcpy.CheckOutExtension("3D")

arcpy.ImportToolbox(r"c:\program
files\arcgis\pro\Resources\ArcToolbox\toolboxes\GeoAnalytics Desktop Tools.tbx.tbx")
DOC_15_ecw = arcpy.Raster("DOC_15.ecw")
MODEL_dlpk =
"D:\Doctorat\3_An_III\5_Teza\4_2_Extragerea_amprentelor_la_sol_ale_constructiilor\1_ArcGIS_Pro_Deep_Learning_Trained\1_V4_dupa_v5\1_V4\MODEL\MODEL.dlpk"

# Process: Detect Objects Using Deep Learning (Detect Objects Using Deep Learning) (ia)
DOC_15_Building_FP =
"D:\Doctorat\3_An_III\5_Teza\4_2_Extragerea_amprentelor_la_sol_ale_constructiilor\1_ArcGIS_Pro_Deep_Learning_implementat\1_V1\1_V1.gdb\DOC_15_Building_FP"
arcpy.ia.DetectObjectsUsingDeepLearning(in_raster=DOC_15_ecw,
out_detected_objects=DOC_15_Building_FP, in_model_definition=MODEL_dlpk,
arguments=[["padding", "128"], ["batch_size", "4"], ["threshold", "0.9"], ["return_bboxes", "False"],
["tile_size", "512"]], run_nms="NO_NMS", confidence_score_field="Confidence",
class_value_field="Class", max_overlap_ratio=0,
processing_mode="PROCESS_AS_MOSAICKED_IMAGE")
.save(Detect_Objects_Using_Deep_Learning)

# Process: Regularize Building Footprint (Regularize Building Footprint) (3d)
DOC_15_Building_FP_Regulariz =
"D:\Doctorat\3_An_III\5_Teza\4_2_Extragerea_amprentelor_la_sol_ale_constructiilor\1_ArcGIS_Pro_Deep_Learning_implementat\1_V1\1_V1.gdb\DOC_15_Building_FP_Regulariz"
arcpy.ddd.RegularizeBuildingFootprint(in_features=DOC_15_Building_FP,
out_feature_class=DOC_15_Building_FP_Regulariz, method="RIGHT_ANGLES", tolerance=1,
densification=None, precision=0.15, diagonal_penalty=1.5, min_radius=0.1, max_radius=1000000,
alignment_feature="", alignment_tolerance="", tolerance_type="DISTANCE")

# Process: Dissolve Boundaries (Dissolve Boundaries) (gapro)
Output_Feature_Class =
"D:\Doctorat\3_An_III\5_Teza\4_2_Extragerea_amprentelor_la_sol_ale_constructiilor\1_ArcGIS_Pro_Deep_Learning_implementat\1_V1\1_V1.gdb\DOC_15_Building_FP_Regulariz_DissolveBoundaries"
arcpy.gapro.DissolveBoundaries(input_layer=DOC_15_Building_FP_Regulariz,
out_feature_class=Output_Feature_Class, multipart="SINGLE_PART", dissolve_fields="", fields=[],
summary_fields=[])

if __name__ == '__main__':
# Global Environment settings
with
arcpy.EnvManager(scratchWorkspace=r"D:\Doctorat\3_An_III\5_Teza\4_2_Extragerea_amprentelor_la_sol_ale_constructiilor\1_ArcGIS_Pro_Deep_Learning_implementat\1_V1\1_V1.gdb",
workspace=r"D:\Doctorat\3_An_III\5_Teza\4_2_Extragerea_amprentelor_la_sol_ale_constructiilor\1_ArcGIS_Pro_Deep_Learning_implementat\1_V1\1_V1.gdb"):
Model()

```

5.2.3. Buildings Footprint Extraction from the LiDAR Point Cloud

To extract buildings footprints from the LiDAR point cloud, I used ArcGIS Pro software. The first step was to apply a filter to the already classified point cloud to only display the points belonging to building class. Next, I created a raster image whose values reflect statistical information about the point cloud. To even out the geometries of the buildings and fill in the gaps in the raster image, I used the *Elevation Void Fill* function. Finally, I obtained the result from Figure 5.27.

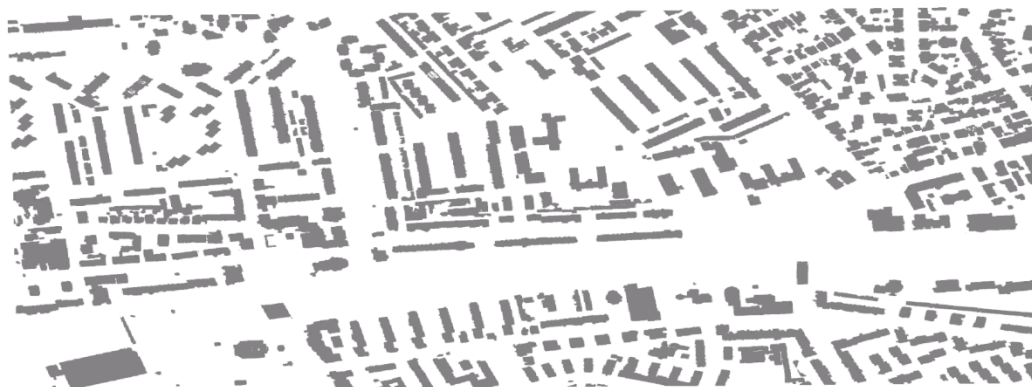


Figure 5.27. Raster Image of Buildings Footprints Extracted from the LiDAR Point Cloud

From the resulted raster image, I extracted the polygons of the buildings footprints. Finally, I corrected the polygons' distortions. I obtained a number of 545 polygons, and the result can be found in Figure 5.28.



Figure 5.28. Buildings Footprints Generated from the LiDAR Point Cloud

5.2.4. Comparative Study and Conclusions

The results obtained with the three methods for extracting buildings footprints can be found in Table 5.2.

Table 5 2. The results obtained with the three methods of extracting the buildings footprints

No.	Method	No. of buildings initially identified	No. of buildings obtained after eliminating overlaps and unifying adjacent buildings
1	Extracting buildings footprints with the DL model from ArcGIS Living Atlas	464	390
2	Extracting buildings footprints with the trained DL model	1263	628
3	Extracting buildings footprints from the LiDAR point cloud	545	545

In the case of using the trained DL model, it was found that a higher number of buildings footprints are obtained than with the model implemented in ArcGIS Pro. This is also due to the fact that there is a better segmentation of the buildings according to the differences noticed in the roofs. Thus, several adjacent blocks were identified as different constructions and not the same construction as when using the DL model implemented in ArcGIS Pro. As a result of joining adjacent geometries and eliminating overlaps, this difference decreased considerably. When extracting buildings footprints from the LiDAR point cloud, since generating them is also based on elevation data, and several adjacent blocks can have the same number of levels and thus the same height, they have been identified as a single building (Figure 5.29).

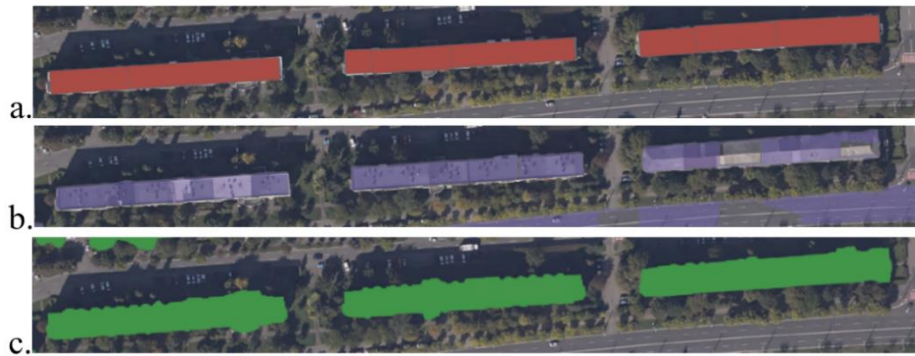


Figure 5.29. Blocks Identified with the DL Model Implemented in ArcGIS Pro (a.), with the Trained DL Model (b.), and based on the LiDAR Point Cloud (c.)

For more complex buildings, such as the church in Figure 5.30, it is found that in the case of the model implemented in ArcGIS Pro, it was not classified as a building, and in the case of using the trained DL model, although identified, the resulting geometry does not correspond to reality, requiring manual intervention. Also, in the case of using the LiDAR point cloud, errors may occur, due to improper classification of points or lack of points in some areas.

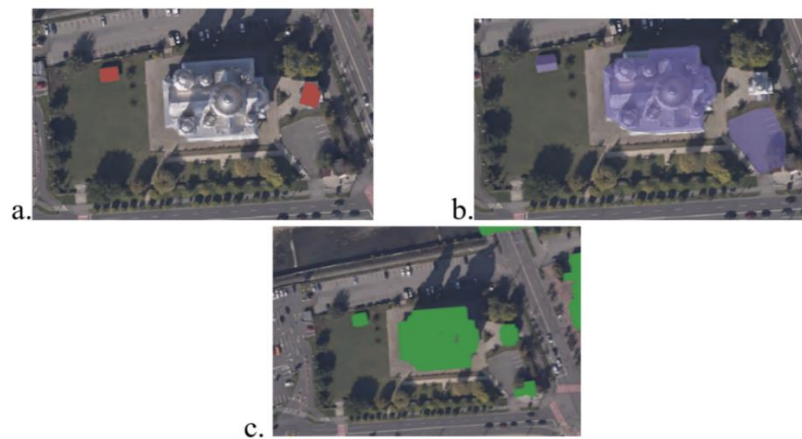


Figure 5.30. Church Identified with the DL Model Implemented in ArcGIS Pro (a.), with the Trained DL Model (b.), and based on the LiDAR Point Cloud (c.)

In image segmentation with the trained DL model, portions of the river were identified as buildings (Figure 5.31), requiring manual intervention to remove them.



Figure 5.31. River Areas Classified as Buildings

In order to more effectively evaluate the performance of the studied models, I made a comparison with the geometry of a building footprint from eTerra. For this, I chose the rectangular construction in Figure 5.32. Analyzing the differences between the coordinates of the 4 corners of the building in Table 5.3 and considering the coordinates of the building footprint in eTerra as the correct value, it can be concluded that the smallest sum of errors was obtained from the trained DL model.

Table 5.3. Comparison of the Coordinates of the Building Footprint Obtained from the 3 Described Methods and the Coordinates of the Corresponding Geometry from eTerra

Coordinates from the DL Model Implemented in ArcGIS Pro			
E [m]	er. [m]	N [m]	er. [m]
391540.970	-0.355	685519.900	0.078
391612.020	-0.791	685527.610	-0.807
391613.170	-0.894	685516.290	-1.703
391542.310	-0.269	685508.510	-0.884
S=	-2.310	S=	-3.316
Coordinates from the Trained DL Model			
E [m]	er. [m]	N [m]	er. [m]
391540.250	-1.075	685520.400	0.578
391613.260	0.449	685529.180	0.763
391614.650	0.586	685517.580	-0.413
391541.640	-0.939	685508.800	-0.594
S=	-0.980	S=	0.334
Coordinates from the LiDAR Point Cloud			
E [m]	er. [m]	N [m]	er. [m]
391540.010	-1.315	685520.610	0.788
391612.830	0.019	685529.580	1.163
391614.480	0.416	685516.260	-1.733
391541.410	-1.169	685508.160	-1.234
S=	-2.050	S=	-1.016
Coordinates from eTerra			
E [m]	N [m]		
391541.325	685519.822		
391612.811	685528.417		
391614.065	685517.993		
391542.579	685509.394		

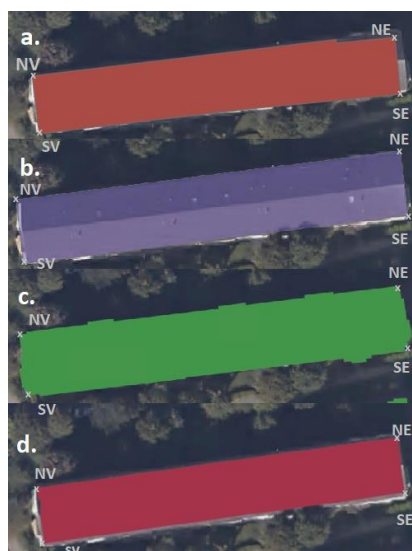


Figure 5.32. Buildings Identified with the DL Model Implemented in ArcGIS Pro (a.), the Trained DL Model (b.), the LiDAR Point Cloud (c.), and eTerra (d.)

5.3. Obtaining the 3D Buildings Model

5.3.1. 3D Modelling of the Buildings in ArcGIS Pro

For the 3D modelling of the buildings in ArcGIS Pro, multipatch objects are used. In this case, I used the buildings footprints obtained from the LiDAR point cloud and the point cloud classified using the automated classification tools implemented in ArcGIS Pro with manual corrections of the classification errors. The 3D model obtained can be found in Figure 5.33.

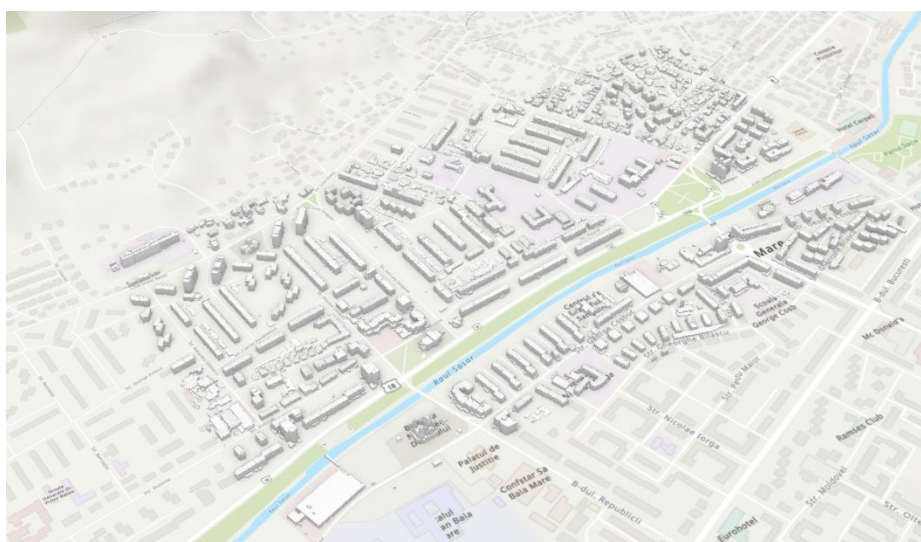


Figure 5.33. 3D Model of the Buildings – Overview

Looking closely at the obtained buildings, I noticed that their surface is irregular (Figure 5.34).

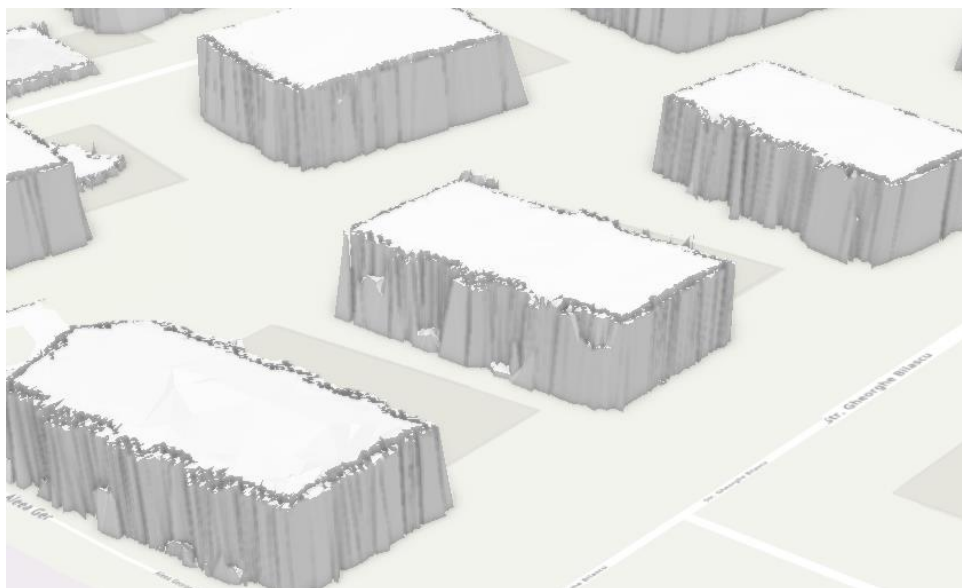


Figure 5.34. 3D Model – Detail View

To solve this problem, I opted for a manual rectification of the buildings footprints, followed by their extrusion using the Extrusion function, at a height determined from the LiDAR point cloud, which was noted in the attribute table of the buildings layer. Thus, 3D models of the buildings with LOD1 were obtained (Figure 5.35).

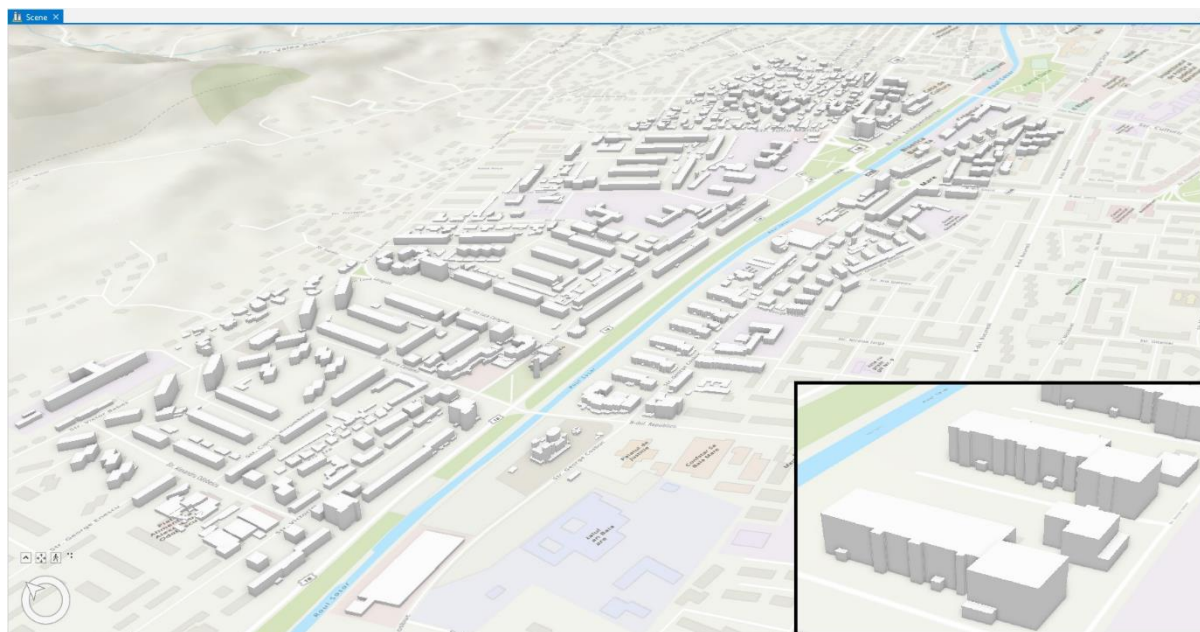


Figure 5.35. 3D Model of the Buildings Obtained with Extrusion in ArcGIS Pro, Representation in LOD1

To represent the buildings in LOD2, roof surfaces for an area were modelled and textures were applied based on a package of rules from CityEngine. In this case, the *Building Shell with Detail* rules package was applied, adding facades with architectural details to a 3D body (Schueren). The modelling result can be found in Figure 5.36.



Figure 5.36. 3D Model of the Buildings – LOD2 Representation

5.3.2. 3D Buildings Modeling in CityEngine

CityEngine is a rule-based urban modelling software package (Kelly, 2021). The programming language used in CityEngine is CGA – Computer-generated Architecture and based on it, *.cga format files are created, that contain rules which are applied to a 2D geometry to obtain the 3D model of the building ("CGA modelling overview– ArcGIS CityEngine Resources | Documentation").

The first step was the generation of streets, for which the axis was vectorized, and then the CGA *CompleteStreets_2022* rules were applied (Wasserman, 2023). These include the rules in Figure 5.37, which divide the street into pavement and lanes, and the rules in Figure 5.38, which give texture to streets. For each street, parameters corresponding to the width of the road surface and sidewalks were chosen. Following the generation of the streets, the lots resulted, to which textures were applied. The results can be found in Figure 5.39.

```
RightSide-->
split(v,unitSpace,0){(_Transit_Lane_Width_Switch(0,"Sidewalk Side")):Transit_Lane_Reporting(0)
|_RightSplitSum:RightSplitSpace
|(_Transit_Lane_Width_Switch(0,"Right Most Lane")):Transit_Lane_Reporting(0)
|(_Distribute_Right_Lanes*_Actual_Lane_Width):Lanes(0,(case rightHandTraffic: Stop_End else: Stop_Begin))
|_Transit_Lane_Width_Switch(0,"Left Most Lane")):Transit_Lane_Reporting(0)}

LeftSide-->
split(v,unitSpace,0){(_Transit_Lane_Width_Switch(2,"Left Most Lane")): scaleUV(0,-1,-1) Transit_Lane_Reporting(2)
|(nLanesLeft*_Actual_Lane_Width ): translateUV(0,0,-geometry.vMax) scaleUV(0,-1,-1) |
Lanes(2,(case rightHandTraffic: Stop_Begin else: Stop_End) )
|(_Transit_Lane_Width_Switch(2,"Right Most Lane")):scaleUV(0,-1,-1) Transit_Lane_Reporting(2)
|_LeftSplitSum:LeftSplitSpace
|_Transit_Lane_Width_Switch(2,"Sidewalk Side")):scaleUV(0,-1,-1) Transit_Lane_Reporting(2)}
```

Figure 5.37. CGA Rule for Dividing Streets into Sidewalks and Lanes

```

@StartRule
Street -->
  set(material.name,"Road_Bed")
  Initial_Street_Settings

Initial_Street_Settings-->
  case Specularity==1 && Transparency==0:
    Short_Long_Street
  else:
    Short_Long_Street_Altered

Short_Long_Street_Altered-->
  case Transparency>0 && Specularity==1:
    set(material.opacity,1-Transparency)
    Short_Long_Street
  else:
    set(material.specular.g,Specularity)
    set(material.specular.b,Specularity)
    set(material.specular.r,Specularity)
    Short_Long_Street

Short_Long_Street-->
  case _uScale==1:
    Long_Street

  else:
    Short_Street

Long_Street-->
  ReportMultimodalMetrics(reportingOn)
  BridgeMain
  split(u,uvSpace,1){_crosswalkBeginWidth/10: Asphalt("Auto","asphalt")
                    |_crosswalkBeginWidth : Crosswalk( Crosswalk_Begin ,1)
                    |~1 : StreetWithCrosswalkEnd }

Short_Street-->
  ReportMultimodalMetrics(reportingOn)
  BridgeMain
  set( Right_Bike_Box ,"false")
  set( Left_Bike_Box ,"false")
  set( Stop_Begin ,"none")
  set( Stop_End ,"none")
  StreetWithEntries

```

Figure 5.38. Extract from the CGA Rule that Gives Texture to the Streets



Figure 5.39. Generated Lots with Applied Textures

In order to obtain the 3D models of the buildings, I imported into CityEngine the shapefile with the buildings footprints generated with the trained DL model and manually corrected, having their height determined from the LiDAR point cloud and mentioned in the attribute table. The CGA rules in Figure 5.40 were applied for the extrusion of the buildings footprints, generation of hip roofs, shaping of building facades, differentiating the front façade, which also includes entrances, from side

ones, division of buildings into ground floor and floors, generation of windows and application of textures and colors. The final result can be found in Figure 5.41.

```

attr Inaltimea_ = 0
attr groundfloor_height = 3.5
attr floor_height = 3
attr tile_width = 4
const window_color = "#a1c2e6"
const frame_color = "#c2c2c2"
const door_color = "#b8b8b8"

@StartRule
Lot -->
  extrude(Inaltimea_)
  Volume
Volume --> comp(f) { front : Frontfacade | side : Sidefacade | top : Top }
Frontfacade -->
  split(y) { groundfloor_height : Groundfloor
            | { ~floor_height : Floor}* }
Top -->
  roofHip(30, 1)
  color("#9e3b2c")

Sidefacade -->
  split(y) { groundfloor_height : Floor
            | { ~floor_height : Floor}* }
Floor -->
  split(x) { 0.4 : SolidWall
            | { ~tile_width : Tile }*
            | 0.4 : SolidWall }
Groundfloor -->
  split(x) { 0.4 : SolidWall
            | { ~tile_width : Tile }*
            | ~tile_width : EntranceTile
            | 0.4 : SolidWall }
Tile -->
  split(x) { 1 : SolidWall
            | ~1 : split(y) { 1.2 : SolidWall | ~1.3 : Window | 0.5 : SolidWall }
            | 1 : SolidWall }
EntranceTile -->
  split(x) { ~1 : SolidWall
            | 2.5 : split(y) { 3 : Door | ~2 : SolidWall }
            | ~1 : SolidWall }
SolidWall -->
  s('1, '1, -0.4)
  primitiveCube()
  color("#f7fad4")

Window -->
  t(0, 0, -0.2)
  split(y) { 0.1 : Frame
            | ~1 : split(x) { 0.1 : Frame | { ~1 : Glass | 0.1 : Frame }* }
            | 0.1 : Frame }
Glass -->
  color(window_color)
  set(material.specular.r, 0.5)
  set(material.specular.g, 0.5)
  set(material.specular.b, 1)
  set(material.reflectivity, 1)
  set(material.shininess, 100)
  set(material.opacity, 0.5)

Frame -->
  color(frame_color)

Door -->
  t(0, 0, -0.3)
  split(y) { ~1 : split(x) { 0.15 : Frame
                          | ~1 : Panel
                          | 0.05 : Frame
                          | ~1 : Panel
                          | 0.15 : Frame }
            | 0.15 : Frame }
Panel -->
  color(door_color)

```

Figure 5. 40. CGA Rules for Building Façades Modelling

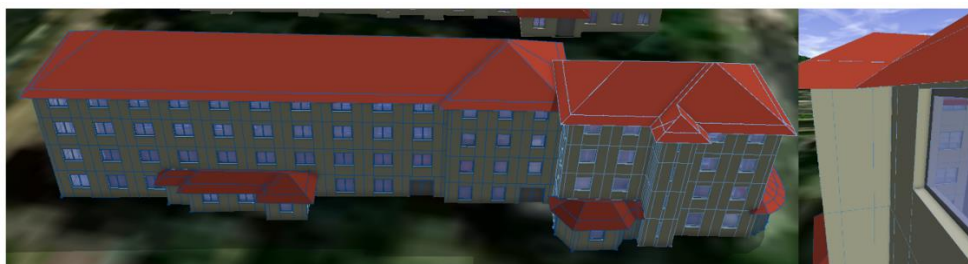


Figure 5.41. Building Modelling in CityEngine in LOD3 (LOD3.1, according to classification by Biljecki et al. (2016))

The same rules, with variations in colours, were applied to the other buildings, obtaining the result in Figures 5.42, 5.43 and 5.44.



Figure 5.42. Area modelling in CityEngine with LOD detail level 3 (LOD3.1, According to the Classification Made by Biljecki et al. (2016)), Overview



Figure 5.43. 3D Model of the Area, Detail View



Figure 5.44. 3D Model of the Area, Detail View

5.3.3. Obtaining the 3D City Model Based on Open-Source Data

I used OpenStreetMap as an open-source data that can be imported into CityEngine. The imported data consists of the road network, buildings footprints and a satellite image. 3D models of the buildings were generated based on imported footprints. To the obtained volumes a package of rules such as those described above were applied, rules designed for buildings modelled based on data from OpenStreetMap, and the 3D model of Baia Mare Municipality can be found in Figure 5.45.

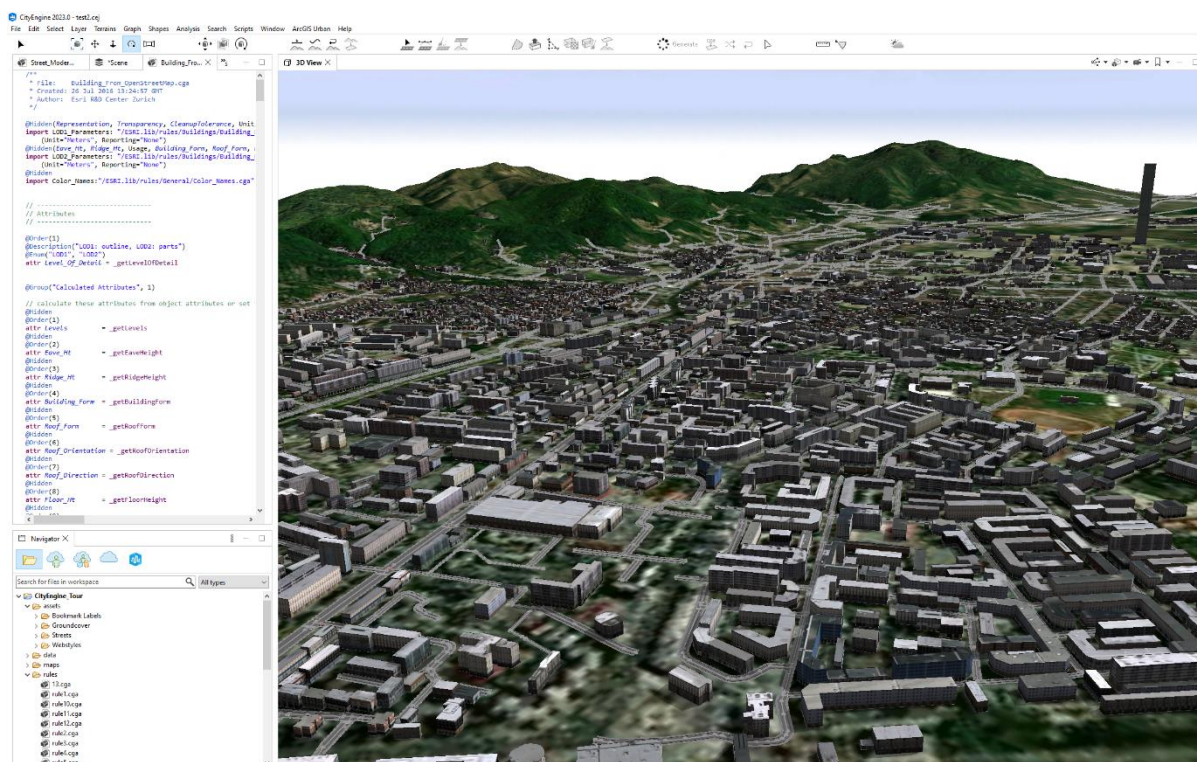


Figure 5.45. 3D Model of Baia Mare Municipality (LOD2)

5.3.4. Visualizing 3D City Models

In order to visualize the 3D model of Baia Mare, I published it as a *hosted scene layer* (Figure 5.46).

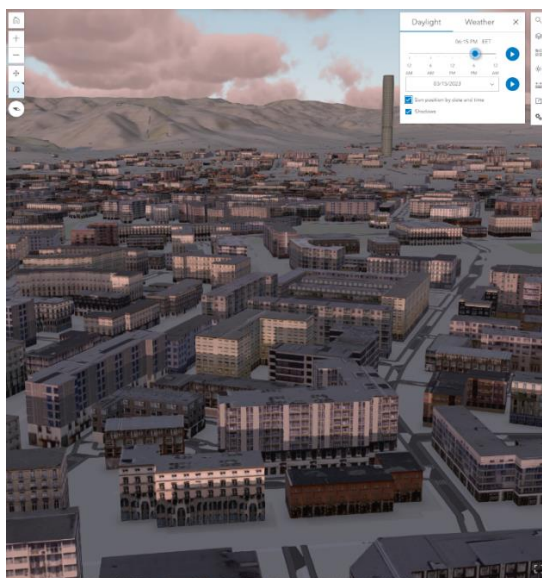


Figure 5.46. 3D Model of Baia Mare in ArcGIS Online, with Setting the Time of the Day

5.3.5. AR and VR Application for the Visualization of 3D City Models

In order to access the 3D model of the city in a VR and AR environment, I created Bookmarks with points of interest, which I exported in a *.3vr format, compatible with virtual reality visualization applications. The created AR environment can be viewed in a browser (Figures 5.47 and 5.48) by rotating the image using the mouse, or on mobile applications (Figures 5.49 and 5.50) by rotating the image by moving the phone. The created VR environment can be accessed through a VR headset, with the user moving around to rotate the image.



Figure 5.47. Visualization of the 3D Model of Baia Mare, Generated Based on Open-source Data, in an AR Environment in a Browser



Figure 5.48. Visualization of the 3D AR Model of an Area in Baia Mare, Generated Based on the LiDAR Point Cloud and Photogrammetric Images

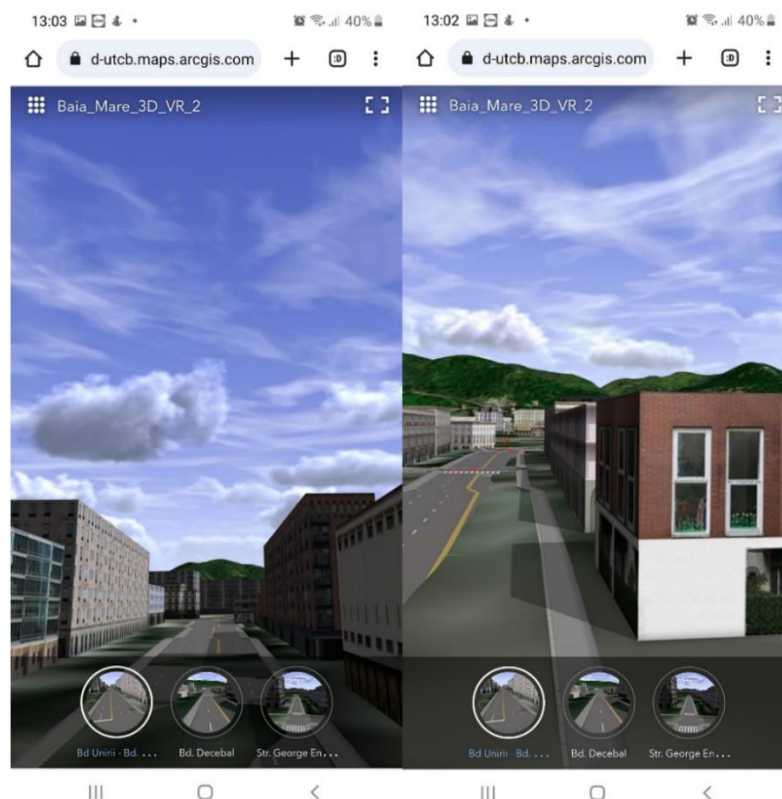


Figure 5.49. Visualization of the 3D model of Baia Mare, Generated Based on Open-source Data, in a Mobile AR Environment



Figure 5.50. Visualization of the 3D AR Model of an Area in Baia Mare, Generated Based on the LiDAR Point Cloud and Photogrammetric Images

HTC Vive Cosmos (Figure 5.51) is a six-camera VR system with a pixel resolution of 2880x1700 and can be connected to a computer to view various VR applications ("VRpro.ro: HTC Vive Cosmos"). This system was used to visualize the 3D model of the city in a VR environment (Figure 5.52).



Figure 5.51. HTC Vive Cosmos VR System ("VRpro.ro: HTC Vive Cosmos")



Figure 5.52. Viewing the 3D Model of Baia Mare, Generated Based on Open-source Data, in a VR Environment, through the HTC Vive Cosmos VR System

5.4. Conclusions

In this chapter, two software products for point cloud classification and a Deep Learning model trained on a dataset from the study area were tested. I found that for the efficient training of such a model, it is necessary for the training dataset to include a very large and diversified volume of data, and the performance of the hardware system is a very important factor. The most effective solution in terms of results and the least need for manual error correction was to use the automatic point cloud classification tools implemented in ArcGIS Pro. This option was better for classifying a larger volume of data such as that needed for a large urban area. Creating an application in the Python programming language has increased the automatization of the entire process and has helped to reduce the time spent classifying the point cloud.

To extract buildings footprints I used ArcGIS Pro software through different methods: using an existing DL model, using the point cloud, and training a DL model for the specifics of the study area. Although all three methods require manual interventions to correct image classification errors, by comparing it with the geometry of a building from eTerra, the trained DL model provided the result that was closest to reality.

Regarding the generation of 3D buildings models, using an algorithm to generate them based on the LiDAR point cloud did not provide the desired results, the surfaces of the resulted models being irregular. Such a method involves major manual interventions to rectify the resulting errors. Thus, I opted to use a function to extrude the buildings footprints, at a height determined from the LiDAR point cloud that was mentioned in their attribute table. I did the same in the CityEngine application, but in this case the modelling was done thoroughly. In this regard, I proposed a set of rules that were applied to the buildings footprints to obtain 3D models of them, including details such as doors and windows at façade level. Such modelling can be used to manage emergency response, where façade elements can be an important factor.

Open-source geospatial data has proven to be sufficient to obtain a 3D model of the city, but its accuracy is low.

The presentation of 3D models of buildings and the possibility of distributing and visualizing them is an important factor. If these models are meant to support society by making resource management and emergency planning more efficient, they must be accessible and understandable to the general public. The distribution of models in a GIS Online environment or their visualization through a computer, a mobile or a VR system in the VR application that was created, can support both citizens and authorities, by reducing on-site trips in situations where this is necessary for the analysis of different scenarios.

6. General Conclusions, Original Contributions and Perspectives

6.1. General Conclusions

3D models of the cities are the basis of thorough analyses that support the improvement of methods for solving the problems that today's society faces and the accuracy with which they are achieved is an important factor in obtaining good predictions in situations such as visibility analysis, energy demand estimation, shadow estimation, energy potential estimation, or sound propagation. Their standardization allows data exchange between different models, and CityGML standards, with its five levels of detail, are widely used today. Depending on the field of application of the 3D model, LOD0, LOD1, or LOD2 can be chosen, for example for visibility analyses, estimation of electricity demand, or energy potential, LOD3, which also includes detailed elements of the walls, roofs and balconies, for shadow estimation or sound propagation analyses, or LOD4, which also encompasses interior building structures, for emergency response management.

The concept of "Smart City" refers to a 3D digital replica of a city, built both on the basis of geospatial data and other dynamic data received in real time, related to the population and how the resources are managed within an urban area.

The main objective of the doctoral thesis was to identify an optimal workflow for generating, distributing and visualizing 3D models of cities, which can be further implemented by authorities or other interested persons. In this regard, I analyzed the main methods of acquiring geospatial data necessary for 3D modelling of the buildings and the main techniques of 3D spatial modelling, the study area being in Baia Mare city.

Depending on the chosen level of detail, the geospatial data acquisition technique will be selected. For levels of detail LOD0, LOD1 and LOD2 data acquisition via airborne laser scanning may be sufficient. For the level of detail LOD3, the accuracy of the generated 3D model depends on the density of the point cloud, and terrestrial laser scanning that allows to obtain a point cloud with a higher density may be more efficient. To generate a model with a level of detail LOD4, terrestrial or mobile laser scanning inside the buildings is required. The acquisition of photogrammetric images ensures the possibility to extract the buildings footprints with greater accuracy than extracting them from the LiDAR point cloud.

For airborne laser scanning, scanners such as Leica CityMapper-2 or RIEGL VQ-780ii-S can be used to obtain the point cloud for a larger area. To obtain a point cloud with a higher density, terrestrial laser scanners such as Leica ScanStation P50 or Trimble TX8 can be used. Mobile laser scanning can be performed through platforms such as the Leica ProScan G-Series or the FGI ROAMER scanning system, developed by the Geodetic Institute of Finland and used for road mapping.

Photogrammetric images can be aerial, taken from an UAV, terrestrial, taken with a camera located on the ground, or satellite. Depending on the angle of retrieval, they can be nadir or inclined. Thus, for the acquisition of aerial photogrammetric images,

nadiral photogrammetric cameras can be used, such as UltraCam Eagle Mark 3 (Vexcel) or Leica DMC II (Leica), and oblique photogrammetric cameras can be used to retrieve oblique photogrammetric images, such as UltraCam Osprey (Vexcel), Leica RCD30 Oblique (Leica) or IGI Quattro DigiCAM Oblique (IGI).

The data for the case study of the doctoral thesis were acquired by airborne laser scanning, with the RIEGL VQ-780ii-S laser scanner, in the case of LiDAR point clouds and with the photogrammetric cameras Phase One P1-iXM-RS150F and P1-iXM-RS100F in the case of photogrammetric images.

LiDAR point cloud classification and buildings footprint extraction are important steps in generating the 3D models of the cities. Recent research in the field is based mainly on machine learning classification with algorithms such as SVM, Adaboost, Random Forest, Markov Random Field or Conditional Random Field (CRF).

The use of DL algorithms to perform these operations is the current standard in the field, and in this regard, convolutional neural networks such as PointCNN, for classifying the LiDAR point cloud and MaskRCNN, for extracting geometric elements from photogrammetric images, stand out.

In the doctoral thesis, for the classification of the LiDAR point cloud, I used the classification model defined by ASPRS, which also includes the classes *Ground*, *Low noise*, *Building*, *Low vegetation*, *Medium vegetation* and *High vegetation*, on which the comparative study within the paper was focused. Different software products may provide different results because they are based on different classification methods, and to that end, I performed point cloud classification with ArcGIS Pro and Bentley Microstation Terrascan software products. Finally, I trained a DL model, based on the convolutional neural network PointCNN, which proved to be effective for its intended purpose. In case of a larger volume of data, which for technical reasons related to the performance of the hardware components that are used, must be divided into several files, it is necessary that the entire process of classifying the LiDAR point clouds to be automated. Thus, I designed a workflow in the Python programming language, which proved to be useful for managing data in a GIS environment.

For extracting the buildings footprints, I used both photogrammetric images and the LiDAR point cloud as starting data. In addition, I used an existing image classification model, taken from ArcGIS Online and also a trained DL model, the latter providing the most realistic results. Through polyline compression algorithms, the distortions of the obtained polygons can be corrected, in order to obtain shapes of the buildings footprints as regular as possible.

Regarding the generation of 3D models of the buildings, performing this operation exclusively based on the point cloud proved to be inefficient, due to irregularities in the point cloud. An optimal solution proved to be the extrusion of the footprints extracted from photogrammetric images, at a height determined from the LiDAR point cloud. Thus, the initial hypothesis was confirmed, namely the need to acquire the two types of geospatial data.

The programming language used in CityEngine – CGA – allows the application of rules on 2D geometries to obtain 3D models of the buildings. The possibilities for modelling are numerous, and in the case study I modelled the buildings with the levels

of detail LOD1, LOD2 and LOD3. Thus, 3D models of the buildings have been generated including façade details, models that can be used to manage the response to emergency situations, in which façade elements such as doors and windows can be an important factor.

Finally, how the modelling output is distributed and visualized is a very important factor, as these models need to be accessible and understandable to the general public. The development of Desktop or Mobile GIS applications was a solution to this problem, and the visualization in a VR environment of the 3D model of the city brings these applications to the current standards and offers the possibility of more efficient management of time and resources by the authorities with responsibilities in the areas where these models can be used.

In conclusion, for the spatial representation of buildings, I proposed the workflow in figure 6.1.

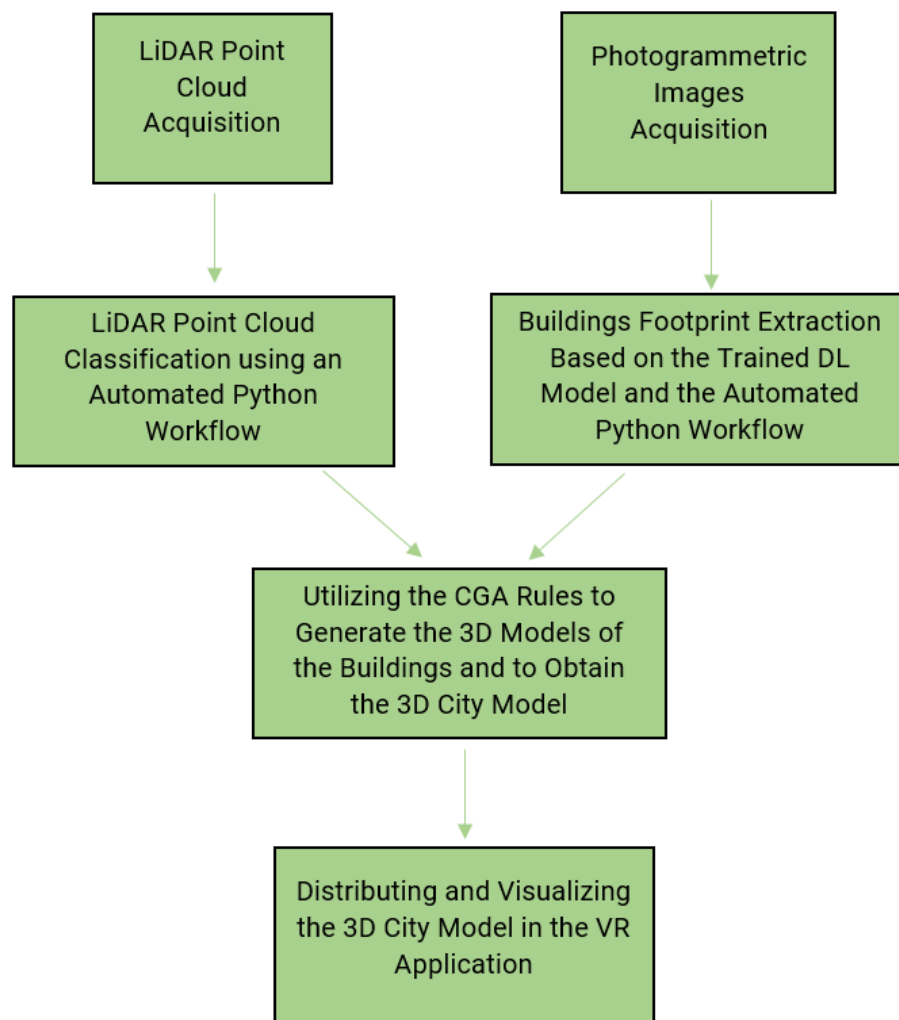


Figure 6.1. Proposed Workflow for Spatial Representation of Buildings

6.2. Original Contributions

My personal contributions were the following:

- Detailed analysis of the main geospatial data acquisition techniques necessary for 3D spatial modelling;
- Analysis of the possibility of using open-source data in 3D city modelling;
- Analysis of current 3D spatial modelling techniques;
- Point cloud classification with ArcGIS Pro software;
- Point cloud classification with Bentley Microstation Terrascan software;
- Training a DL model for classifying the point cloud and implementing it;
- Making a proposal of a workflow for LiDAR point cloud classification automation using the Python programming language;
- Comparative study of point cloud classification methods;
- Extracting buildings footprints in ArcGIS Pro software from photogrammetric images;
- Extracting buildings footprints in ArcGIS Pro software from the LiDAR point cloud;
- Training a DL model for extracting buildings footprints and implementing it;
- Making a proposal of a workflow for extracting buildings footprints automation using the Python programming language;
- Comparative study of the methods used to extract buildings footprints;
- 3D spatial modelling of an area in ArcGIS Pro software;
- Proposing a set of rules in the CGA programming language for 3D spatial modelling of an area;
- 3D spatial modelling of an area in the CityEngine software product;
- Generating 3D models of the buildings in Baia Mare based on available open-source data;
- Creating an ArcGIS Online application to visualize the 3D model of the city;
- Creating a VR application to visualize the 3D model of the city;
- Proposing a workflow for spatial representation of the buildings.

6.3. Perspectives

In order to continue the study in the doctoral thesis, the following can be achieved:

- Improving the trained DL models by expanding the training data areas;
- Study on improving the methods for generating 3D models of buildings based on LiDAR point clouds and identifying a method to smoothen the resulting surfaces;
- Improving CGA modelling rules by creating different rules for different types of buildings;
- 3D spatial modelling of the entire Baia Mare municipality based on data acquired through the technologies presented in the doctoral thesis;
- Completing the 3D model of Baia Mare Municipality with data related to population and resource management, to contribute to the achievement of a Smart City.

References

- 3D BAG Viewer. Available at <https://3dbag.nl/en/viewer?lod=lod22&rdx=154956.97283040546&rdy=462911.66065216874&ox=53.01514268013125&oy=274.55557689257734&oz=-277.67738732388534> (accessed on 12.07.2023).
- About 3D | Boston Planning & Development Agency. Available at <http://www.bostonplans.org/3d-data-maps/3d-smart-model/about-3d> (accessed on 27.05.2022).
- About Linear Regression | IBM. Available at <https://www.ibm.com/topics/linear-regression> (accessed on 14.07.2023).
- About OpenStreetMap - OpenStreetMap Wiki. Available at https://wiki.openstreetmap.org/wiki/About_OpenStreetMap (accessed on 07.09.2023).
- Art Collection. The Metropolitan Museum of Art. Available at <https://www.metmuseum.org/art/the-collection> (accessed on 13.07.2023).
- Attaran, M., Celik, B.G., 2023. Digital Twin: Benefits, use cases, challenges, and opportunities, *Decision Analytics Journal* 6, 100165. <https://doi.org/10.1016/j.dajour.2023.100165>.
- Automate Building Footprint Extraction using Deep learning. ArcGIS API for Python. Available at <https://developers.arcgis.com/python/samples/automate-building-footprint-extraction-using-instance-segmentation/> (accessed on 14.07.2023).
- Badea, A.C., Badea, G., 2022. Open Geospatial Data and Tools for Sustainable Cities - Advantages and Disadvantages, FIG Congress 2022, Warsaw, Poland.
- Biljecki, F., Ito, K., 2021. Street view imagery in urban analytics and GIS: A review. *Landscape and Urban Planning* 215, 104217. <https://doi.org/10.1016/j.landurbplan.2021.104217>.
- 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>.
- 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>.
- 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, 2842–2889. <https://doi.org/10.3390/ijgi4042842>.
- 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. *European COST Action TU0801*, 1-118. <https://doi.org/10.1051/TU0801/201400001>
- Building Footprint Extraction - USA - Overview. Available at <https://sd-utcb.maps.arcgis.com/home/item.html?id=a6857359a1cd44839781a4f113cd5934> (accessed on 12.04.2022).
- Buildings. Available at <https://terrasolid.com/guides/tscan/crbuildings.html> (accessed on 25.04.2023).

- By height from ground. Available at <https://terrasolid.com/guides/tscan/crbyheightfromground.html> (accessed on 25.04.2023).
- Cai, M., Kassens-Noor, E., Zhao, Z., Colbry, D., 2023. Are smart cities more sustainable? An exploratory study of 103 U.S. cities. *Journal of Cleaner Production* 416, 137986. <https://doi.org/10.1016/j.jclepro.2023.137986>.
- Cecotti, H., 2022. Cultural Heritage in Fully Immersive Virtual Reality. *Virtual Worlds* 1, 82–102. <https://doi.org/10.3390/virtualworlds1010006>.
- CGA modelling overview—ArcGIS CityEngine Resources | Documentation. Available at <https://doc.arcgis.com/en/cityengine/2020.1/help/help-cga-modelling-overview.htm> (accessed on 07.09.2023).
- Change LAS Class Codes (3D Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/3d-analyst/change-las-class-codes.htm> (accessed on 25.04.2023).
- Creating a street asset inventory from point clouds by using AI | PointlyAI. Available at <https://pointly.ai/use-case/creating-a-sign-and-street-asset-inventory-from-point-clouds-by-using-ai/> (accessed on 14.07.2023).
- The automated labeling of point clouds from highway scans. 3D point cloud classification made smart, fast and easy | PointlyAI. Available at <https://pointly.ai/use-case/the-automated-labeling-of-point-clouds-from-highway-scans/> (accessed on 14.07.2023).
- The new way of generating BIM models from 3D point clouds | PointlyAI. Available at <https://pointly.ai/use-case/the-new-way-of-generating-bim-model-from-3d-point-clouds/> (accessed on 14.07.2023).
- CityGML | OGC. Available at <https://www.ogc.org/standards/citygml> (accessed on 18.02.2022).
- Citywide 3D Model | Boston Planning & Development Agency. Available at <http://www.bostonplans.org/3d-data-maps/3d-smart-model/citywide-3d-model> (accessed on 12.07.2023).
- Classify LAS Building (3D Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/3d-analyst/classify-las-building.htm> (accessed on 25.04.2023).
- Classify LAS By Height (3D Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/3d-analyst/classify-las-by-height.htm> (accessed on 25.04.2023).
- Classify LAS Ground (3D Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/2.9/tool-reference/3d-analyst/classify-las-ground.htm> (accessed on 10.12.2022).
- crowdAI Mapping Challenge: Baseline, 2023. Available at <https://github.com/crowdAI/crowdai-mapping-challenge-mask-rcnn> (accessed on 14.07.2023).
- Deep Learning with ArcGIS Pro Tips & Tricks: Part 2. ArcGIS Blog. Available at <https://www.esri.com/arcgis-blog/products/arcgis-pro/imagery/deep-learning-with-arcgis-pro-tips-tricks-part-2/> (accessed on 01.05.2023).
- Descărcare LAKI MNT. Available at <https://geoportal.ancpi.ro/portal/apps/webappviewer/index.html?id=3f34ee5af71c400396dda574f0d53274> (accessed on 14.07.2023).

- Descarcare TopRO50.Available at <https://geoportal.ancpi.ro/portal/apps/webappviewer/index.html?id=6fcbd33e7ca64215985b478e94cbab64> (accessed on 14.07.2023).
- Detect Objects Using Deep Learning (Image Analyst)—ArcGIS Pro | Documentation.Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/image-analyst/detect-objects-using-deep-learning.htm> (accessed on 01.05.2023).
- Deuber, M., Cavegn, S., Nebiker, S., 2014. Dense Image Matching - Performance Analysis on Oblique Imagery. GIM International 28, 23–25.
- Export Training Data For Deep Learning (Image Analyst)—ArcGIS Pro | Documentation.Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/image-analyst/export-training-data-for-deep-learning.htm> (accessed on 01.05.2023).
- Fields.Available at <https://terrasolid.com/guides/tscan/mwfields.html> (accessed on 25.04.2023).
- Geography Markup Language. Open Geospatial Consortium.Available at <https://www.ogc.org/standard/gml/> (accessed on 12.07.2023).
- Goad, C.C., 1991. The Ohio State University Highway Mapping Project: The Positioning Component. Proceedings of the 47th Annual Meeting of The Institute of Navigation (1991), pp. 117–120.
- Goel, S., Kealy, A., Lohani, B., 2018. Development and Experimental Evaluation of a Low-Cost Cooperative UAV Localization Network Prototype. Journal of Sensor and Actuator Networks 7, 42. <https://doi.org/10.3390/jsan7040042>.
- Grădinaru, A.P., Badea, A.C., Dragomir, P.I., 2022a. Classifying LiDAR Point Cloud Using Bentley Microstation Terrasolid and Arcgis Pro Software Products, RevCAD 32/2022, 123-136.
- Grădinaru, A.P., Badea, A.C., Dragomir, P.I., 2022b. Using GIS Tools to Analyse Emergency and Civil Protection Situations Specific Issues, RevCAD 32/2022, 51-58.
- Grădinaru, A.P., 2022. Raport de cercetare II - Studiul tehnicilor de modelare spațială, Școala Doctorală a Universității Tehnice de Construcții, București.
- Grădinaru, A.P., Badea, A.C., Dragomir, P.I., Badea, G., 2023. Integrating Cadastral Data with Seismic Risk Data in an Online Building Database for the Historical Centre of Bucharest City. Land 12, 1594. <https://doi.org/10.3390/land12081594>.
- Gribov, A., 2019. Optimal Compression of a Polyline While Aligning to Preferred Directions, International Conference on Document Analysis and Recognition Workshops (ICDARW), 98–102. <https://doi.org/10.1109/ICDARW.2019.00022>.
- Gröger, G., Kolbe, T.H., Czerwinski, A., 2006. Candidate OpenGIS® CityGML Implementation Specification (City Geography Markup Language). Open Geospatial Consortium Inc.
- Ground.Available at <https://terrasolid.com/guides/tscan/crground.html> (accessed on 25.04.2023).
- Guler, D., Yomralioglu, T., 2022. Reviewing the literature on the tripartite cycle containing digital building permit, 3D city modelling, and 3D property ownership. Land Use Policy 121, 106337. <https://doi.org/10.1016/j.landusepol.2022.106337>.

- Haring, A., 2007. Die Orientierung von Laserscanner- und Bilddaten bei der fahrzeuggestützten Objekterfassung. Technical University of Vienna.
- He, K., Gkioxari, G., Dollár, P., Girshick, R., 2018. Mask R-CNN. <https://doi.org/10.48550/arXiv.1703.06870>.
- He, N., Li, G., 2021. Urban neighbourhood environment assessment based on street view image processing: A review of research trends. *Environmental Challenges* 4, 100090. <https://doi.org/10.1016/j.envc.2021.100090>.
- Helsingin 3D-mallit. Available at <https://kartta.hel.fi/3d/mesh/> (accessed on 27.05.2022).
- Helsinki 3D (2023). City of Helsinki. Available at <https://www.hel.fi/en/decision-making/information-on-helsinki/maps-and-geospatial-data/helsinki-3d> (accessed on 12.07.2023).
- Home - IGI - Integrated Geospatial Innovations. Available at <https://www.igi-systems.com/home.html> (accessed on 13.07.2023).
- Hosted layers—Portal for ArcGIS | Documentation for ArcGIS Enterprise. Available at <https://enterprise.arcgis.com/en/portal/11.0/use/hosted-web-layers.htm> (accessed on 08.09.2023).
- Huisman, O., de By, R., 2009. Principles of geographic information systems: an introductory textbook, The International Institute for Geo-Information Science and Earth Observation, Enschede.
- Imobile eTerra - Public. Available at <https://geoportal.ancpi.ro/imobile.html> (accessed on 18.07.2022).
- iXM-RS150F Camera, Phase One Geospatial. Available at <https://geospatial.phaseone.com/cameras/ixm-rs150f/> (accessed on 26.05.2023).
- Jog, S., Dixit, M., 2016. Supervised classification of satellite images, in Conference on Advances in Signal Processing (CASP), 93–98. 10.1109/CASP.2016.7746144.
- Jovanović, D., Milovanov, S., Ruskovski, I., Govedarica, M., Sladić, D., Radulović, A., Pajić, V., 2020. Building Virtual 3D City Model for Smart Cities Applications: A Case Study on Campus Area of the University of Novi Sad. *ISPRS International Journal of Geo-Information* 9, 476. <https://doi.org/10.3390/ijgi9080476>.
- Kattenborn, T., Leitloff, J., Schiefer, F., Hinz, S., 2021. Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing* 173, 24–49. <https://doi.org/10.1016/j.isprsjprs.2020.12.010>.
- Kelly, T., 2021. CityEngine: An Introduction to Rule-Based Modelling, Urban Informatics, The Urban Book Series. Springer, Singapore, 637–662. https://doi.org/10.1007/978-981-15-8983-6_35.
- Kolbe, T., Gröger, G., Plümer, L., 2005. CityGML - Interoperable access to 3D city models, *Geo-information for Disaster Management*, Springer, Delft, 883–889. https://doi.org/10.1007/3-540-27468-5_63.
- Kukko, A., Andrei, C.-O., Salminen, V.-M., Kaartinen, H., Chen, Y., Rönholm, P., Hyypä, H., Hyypä, J., Chen, R., Haggrén, H., Kosonen, I., Čapek, K., 2006. Road Environment Mapping System of the Finnish Geodetic Institute—FGI Roamer. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 36, 241–247.

- 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, 43–61. <https://doi.org/10.1007/s41064-020-00095-z>.
- LAS Building Multipatch (3D Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/3d-analyst/las-building-multipatch.htm> (accessed on 02.09.2023).
- LAS Point Statistics As Raster (Data Management)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/las-point-statistics-as-raster.htm> (accessed on 18.07.2023).
- Leica Geosystems. Available at <https://leica-geosystems.com/> (accessed on 13.07.2023).
- Lemmens, M., 2011. Terrestrial Laser Scanning, In: *Geo-information. Geotechnologies and the Environment* 5. Springer, Dordrecht, 101–121. https://doi.org/10.1007/978-94-007-1667-4_6.
- Li, B., Luo, Z., Mao, B., 2022. Non-photorealistic Visualization of 3D City Models using Visual Variables in Virtual Reality Environments. *Procedia Computer Science*, 9th International Conference on Information Technology and Quantitative Management 214, 1516–1521. <https://doi.org/10.1016/j.procs.2022.11.338>.
- Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B., 2018. PointCNN: Convolution On X-Transformed Points, *Advances in Neural Information Processing Systems* 31. <https://doi.org/10.48550/arXiv.1801.07791>.
- Li, Z., Chen, J., Baltsavias, E., 2008. *Advances in Photogrammetry, Remote Sensing and Spatial Information Sciences: 2008 ISPRS Congress Book*. CRC Press, Beijing.
- Liu, J., Chen, N., Chen, Z., Xu, L., Du, W., Zhang, Y., Wang, C., 2022. Towards sustainable smart cities: Maturity assessment and development pattern recognition in China. *Journal of Cleaner Production* 370, 133248. <https://doi.org/10.1016/j.jclepro.2022.133248>
- Lv, Z., Li, X., Zhang, B., Wang, W., Zhu, Y., Hu, J., Feng, S., 2016. Managing Big City Information Based on WebVRGIS. *IEEE Access* 4, 407–415. <https://doi.org/10.1109/ACCESS.2016.2517076>.
- Upadhyay, N., 2013. Available at https://www.gisresources.com/basic-of-photogrammetry_2/ (accessed on 13.07.2023).
- Miller, C., Abdelrahman, M., Chong, A., Biljecki, F., Quintana, M., Frei, M., Chew, M., Wong, D., 2021. The Internet-of-Buildings (IoB) -Digital twin convergence of wearable and IoT data with GIS/BIM. *Journal of Physics Conference Series* 2042, 12041. <https://doi.org/10.1088/1742-6596/2042/1/012041>
- Multipatches—ArcMap | Documentation. Available at <https://desktop.arcgis.com/en/arcmap/latest/extensions/3d-analyst/multipatches.htm> (accessed on 03.09.2023).
- National Institute of Building Sciences buildingSMART alliance, 2015. *National BIM Standard - United States® Version 3*. Available at <https://www.nibs.org/bimc> (accessed on 03.09.2023).
- Oberst, J., Gwinner, K., Preusker, F., 2014. Exploration and Analysis of Planetary Shape and Topography Using Stereophotogrammetry. *The Encyclopedia of the Solar*

- System, Third Edition, Elsevier, 1223–1233. <https://doi.org/10.1016/B978-0-12-415845-0.00057-8>
- Overview - 3D BAG. Available at <https://docs.3dbag.nl/en/> (accessed on 02.09.2022).
- Overview of the ArcGIS API for Python. ArcGIS API for Python. Available at <https://developers.arcgis.com/python/guide/overview-of-the-arcgis-api-for-python/> (accessed on 02.09.2023).
- P2006T By Tecnam: Affordable Twin Engine Aircraft for Sale, Tecnam Aircraft. Available at <https://tecnam.com/aircraft/p2006t/> (accessed on 26.05.2023).
- Pang, H.E., Biljecki, F., 2022. 3D building reconstruction from single street view images using deep learning. *International Journal of Applied Earth Observation and Geoinformation* 112, 102859. <https://doi.org/10.1016/j.jag.2022.102859>.
- Pârvu, I.M., Picu, I.A.C., Dragomir, P.I., Poli, D., 2020. Urban Classification from Aerial and Satellite Images. *Journal of Applied Engineering Sciences* 10, 163–172. <https://doi.org/10.2478/jaes-2020-0024>.
- Pepe, M., Fregonese, L., Scaioni, M., 2018. Planning airborne photogrammetry and remote sensing missions with modern platforms and sensors. *European Journal of Remote Sensing* 51, 412-436. <https://doi.org/10.1080/22797254.2018.1444945>.
- Point cloud classification using PointCNN. ArcGIS API for Python. Available at <https://developers.arcgis.com/python/guide/point-cloud-segmentation-using-pointcnn/> (accessed on 25.04.2023).
- Population growth. Understanding Global Change. Available at <https://ugc.berkeley.edu/background-content/population-growth/> (accessed on 08.09.2023).
- Prepare Point Cloud Training Data (3D Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/3d-analyst/prepare-point-cloud-training-data.htm> (accessed on 25.04.2023).
- Products and Solutions | Trimble Geospatial. Available at <https://geospatial.trimble.com/products-and-solutions> (accessed on 13.07.2023).
- Quintero, M.S., Van Genechten, B., De Bruyne, M., Poelman, R., Hankar, M., Barnes, S., Caner, H., Budei, L., Heine, E., Reiner, H., Lerma García, J.L., Biosca Taronger, J.M., 2008. Theory and practice on Terrestrial Laser Scanning. Universidad Politécnica de Valencia, Valencia.
- Raster to Polygon (Conversion)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/conversion/raster-to-polygon.htm> (accessed on 18.07.2023).
- Regularize Building Footprint (3D Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/3d-analyst/regularize-building-footprint.htm> (accessed on 01.05.2023).
- Rezultate definitive: Caracteristici demografice – Recensământul Populației și Locuintelor. Available at <https://www.recensamantromania.ro/rezultate-rpl-2021/rezultate-definitive-caracteristici-demografice/> (accessed on 08.09.2023).
- RIEGL - Produktdetail. Available at <http://www.riegl.com/nc/products/airborne-scanning/produktdetail/product/scanner/72/> (accessed on 26.05.2023).

- Rohil, M.K., Ashok, Y., 2022. Visualization of urban development 3D layout plans with augmented reality. *Results in Engineering* 14, 100447. <https://doi.org/10.1016/j.rineng.2022.100447>.
- Rupnik, E., Nex, F., Remondino, F., 2014. Oblique Multi-Camera Systems - Orientation and Dense Matching Issues. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XL-3/W1. <https://doi.org/10.5194/isprsarchives-XL-3-W1-107-2014>.
- Samadzadegan, F., Ramzi, P., Schenk, T., 2008. Aerial Triangulation of Digital Aerial Imagery Using Hybrid Features. *The international Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 37.
- Schrotter, G., Hürzeler, C., 2020. The Digital Twin of the City of Zurich for Urban Planning. *PFG - Journal of Photogrammetry, Remote Sensing and Geoinformation Science* 88, 99–112. <https://doi.org/10.1007/s41064-020-00092-2>.
- Schueren, M.. CityEngine Rule of the Week. ArcGIS Blog. Available at <https://www.esri.com/arcgis-blog/products/3d-gis/3d-gis/cityengine-rule-of-the-week/> (accessed on 04.09.2023).
- Seitz, P., 2007. Photon-noise limited distance resolution of optical metrology methods, in: *Optical Measurement Systems for Industrial Inspection V*, SPIE, 109–118. <https://doi.org/10.1117/12.732040>.
- Soininen, A., 2022. TerraScan USER GUIDE.
- Solar Energy Potential. Available at <https://kartta.hel.fi/3d/solar/#/> (accessed on 27.05.2022).
- Sun, T., Xu, Z., Yuan, J., Liu, C., Ren, A., 2017. Virtual Experiencing and Pricing of Room Views Based on BIM and Oblique Photogrammetry, *Procedia Engineering* 196, 1122-1129. <https://doi.org/10.1016/j.proeng.2017.08.071>.
- Tang, L., Ying, S., Li, L., Biljecki, F., Zhu, H., Zhu, Y., Yang, F., Su, F., 2020. An application-driven LOD modelling paradigm for 3D building models. *ISPRS Journal of Photogrammetry and Remote Sensing* 161, 194–207. <https://doi.org/10.1016/j.isprsjprs.2020.01.019>.
- Tao, V., Li, J., 2007. *Advances in Mobile Mapping Technology*, 1st edition. ISPRS Book Series. Taylor&Francis, London.
- Terrasolid UAV. Available at <https://terrasolid.com/products/terrasolid-uav/> (accessed on 14.07.2023).
- Train Deep Learning Model (Image Analyst)—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/tool-reference/image-analyst/train-deep-learning-model.htm> (accessed on 01.05.2023).
- Use ModelBuilder—ArcGIS Pro | Documentation. Available at <https://pro.arcgis.com/en/pro-app/latest/help/analysis/geoprocessing/modelbuilder/modelbuilder-quick-tour.htm> (accessed on 02.09.2023).
- Vexcel Imaging. Vexcel Imaging. Available at <https://www.vexcel-imaging.com/> (accessed on 13.07.2023).
- Virtual Singapore, 2019. Dassault Systèmes. Available at <https://www.3ds.com/insights/customer-stories/virtual-singapore> (accessed on 12.07.2023).
- Virtual Singapore. Available at <https://www.sla.gov.sg/geospatial/gw/virtual-singapore> (accessed on 12.07.2023).

- VirtualcitySYSTEMS, 2017a. Available at <https://kartta.hel.fi/3d/#/> (accessed on 27.05.2022).
- VirtualcitySYSTEMS, 2017b. Available at <https://kartta.hel.fi/3d/atlas/#/> (accessed on 27.05.2022).
- Vosselman, G., Maas, H.-G., 2010. Airborne and Terrestrial Laser Scanning. Whittles Publishing, Dunbeath.
- VRpro.ro: HTC Vive Cosmos. Available at <https://www.vrpro.ro/vive-cosmos> (accessed on 08.09.2023).
- Wang, H., Chen, X., Jia, F., Cheng, X., 2023. Digital twin-supported smart city: Status, challenges and future research directions. *Expert Systems with Applications* 217, 119531. <https://doi.org/10.1016/j.eswa.2023.119531>.
- Wang, W., Lv, Z., Li, X., Xu, W., Zhang, B., Zhang, X., 2015. Virtual Reality Based GIS Analysis Platform, in: Arik, S., Huang, T., Lai, W.K., Liu, Q. (Eds.), *Neural Information Processing, Lecture Notes in Computer Science*. Springer International Publishing, Cham, 638–645. https://doi.org/10.1007/978-3-319-26535-3_73.
- Wang, W., Lv, Z., Li, X., Xu, W., Zhang, B., Zhu, Y., Yan, Y., 2018. Spatial query based virtual reality GIS analysis platform. *Neurocomputing, Query Understanding* 274, 88–98. <https://doi.org/10.1016/j.neucom.2016.06.099>.
- Wasserman, D., 2023. What is the Complete Street Rule? Available at https://github.com/d-wasserman/Complete_Street_Rule (accessed on 13.09.2023).
- Wei, G., He, B.-J., Sun, P., Liu, Y., Li, R., Ouyang, X., Luo, K., Li, S., 2023. Evolutionary trends of urban expansion and its sustainable development: Evidence from 80 representative cities in the belt and road initiative region. *Cities* 138, 104353. <https://doi.org/10.1016/j.cities.2023.104353>.
- Weil, C., Bibri, S.E., Longchamp, R., Golay, F., Alahi, A., 2023. Urban Digital Twin Challenges: A Systematic Review and Perspectives for Sustainable Smart Cities. *Sustainable Cities and Society* 99, 104862. <https://doi.org/10.1016/j.scs.2023.104862>.
- Wen, C., Li, X., Yao, X., Peng, L., Chi, T., 2021. Airborne LiDAR point cloud classification with global-local graph attention convolution neural network. *ISPRS Journal of Photogrammetry and Remote Sensing* 173, 181–194. <https://doi.org/10.1016/j.isprsjprs.2021.01.007>.
- What are Convolutional Neural Networks? | IBM. Available at <https://www.ibm.com/topics/convolutional-neural-networks> (accessed on 14.07.2023).
- What are Neural Networks? | IBM. Available at <https://www.ibm.com/topics/neural-networks> (accessed on 14.07.2023).
- What Is BIM | Building Information Modelling | Autodesk. Available at <https://www.autodesk.com/industry/aec/bim> (accessed on 12.07.2023).
- What's the Difference Between AR and VR? 2020. Tulane School of Professional Advancement. Available at <https://sopa.tulane.edu/blog/whats-difference-between-ar-and-vr> (accessed on 13.07.23).
- Wolf, P.R., Dewitt, B.A., Wilkinson, B.E., 2014. *Elements of Photogrammetry with Applications in GIS*, 4th Edition. McGraw-Hill Education.
- Xu, J., Shu, X., Qiao, P., Li, S., Xu, J.-G., 2023a. Developing a digital twin model for monitoring building structural health by combining a building information

- model and a real-scene 3D model. *Measurement* 217, 112955. <https://doi.org/10.1016/j.measurement.2023.112955>.
- Xu, J., Wang, C., Li, S., Qiao, P., 2023b. Emergency Evacuation Shelter Management and Online Drill Method Driven by Real Scene 3d Model. *SSRN Electronic Journal*. <https://doi.org/10.2139/ssrn.4343148>
- Zhang, C., Lu, Y., 2021. Study on artificial intelligence: The state of the art and future prospects. *Journal of Industrial Information Integration* 23, 100224. <https://doi.org/10.1016/j.jii.2021.100224>.
- Zhu, L., Hyypä, J., Kukko, A., Kaartinen, H., Chen, R., 2011. Photorealistic Building Reconstruction from Mobile Laser Scanning Data. *Remote Sensing* 3, 1406–1426. <https://doi.org/10.3390/rs3071406>.