# Deep Learning Aided Web-Based Procedural Modelling of LOD2 City Models

**Ziya Usta** (ziyausta@artvin.edu.tr)
Artvin Coruh University
**Alper Tunga Akın**
Karadeniz Technical University
**Çetin Cömert**
Karadeniz Technical University

---

**Research Article**

**Keywords:** 3D Modelling, City Models, Aerial Imagery, Procedural Modelling, Deep Learning

**Posted Date:** March 31st, 2023

**DOI:** https://doi.org/10.21203/rs.3.rs-2732563/v1



**Additional Declarations:** No competing interests reported.

---

**Version of Record:** A version of this preprint was published at Earth Science Informatics on July 29th, 2023. See the published version at https://doi.org/10.1007/s12145-023-01053-0.

# Abstract

Use cases such as shadow or solar potential analysis require the use of the LOD2 building models (Level of Detail 2) and the generation of the LOD2 models requires the proper generation of the roof geometries. In general, obtaining roof type information and succeeding generations of the LOD2 models requires expensive aerial surveys and time-consuming construction processes. In this study, a methodology to generate LOD2 building models using only 2D building footprints and aerial imagery is explained to overcome these challenges. Using this methodology, condominiums could be generated as 3D if condominium unit plans are provided as well. The roof type information has been obtained from an aerial image that covers the entire study area using a CNN (Convolutional Neural Network) model with an 89.9 % accuracy rate. Then, the roof geometries have been constructed procedurally by extending and implementing the Straight Skeleton (SS) algorithm for three main types of roofs: flat, gable and hipped. These constructed roof geometries have been combined with LOD1 block models generated by extruding the 2D footprints according to the height attribute. The proposed methodology has been developed as a web-based solution utilizing RESTful web services with modern web technologies. In summary, the main novelty of the study is based on two contributions: the extension of the SS algorithm for the construction of roof geometries and the web-based generation of LOD2 building models.


# 1. Introduction

With the rapid development in data acquisition methods and computer graphics, 3D City Models (3DCM) are widely used, mainly in the city planning industry. A literature review about 3DCMs and their applications can be found in [1]. With the spread of open data policies, which are based on institutions making their data publicly available [2], many institutions in Europe and the United States share 3DCMs as open data via the web using web technologies HTML5 and WebGL. Rotterdam, Amsterdam, Delft, Berlin, and New York are notable ones.

Most of the 3DCMs are found as LOD1 block models in practice since LOD2 models are much more difficult to obtain because of the need for time-consuming processes and expensive data acquisition techniques [3]. The minimum required level of detail in 3DCMs varies according to use cases [4]. While LOD1 models are sufficient for some use cases such as shadow analysis, other use cases require general roof geometries, such as solar analysis [5, 6].

In this work, the possibility of constructing LOD2 models using 2D data such as building footprints and aerial images has been investigated. It is exceedingly difficult to model building roofs from 2D building footprints without roof type information. Deep learning techniques (DL) have been used to derive roof types from aerial imagery to overcome this issue.

DL is a machine learning technique that a computer model learns to perform classification tasks directly from images, text, or sound. DL models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labelled data and neural network architectures that contain many layers [7]. Thus, a DL model called CNN (Convolutional Neural Network) has been used to classify three main roof types with an aerial image in this work. After the classification, extracted roof type information has been used to construct roof geometries. The three main roof types are Hipped, Gable and Flat.

The generation of 3DCMs as LOD2 is done using procedural modelling techniques (PM) in general [8, 9]. PM is an umbrella term which includes many modelling techniques such as L-Systems, fractals, and shape grammar. All these techniques aim to create multiple instance models by utilizing a set of predefined rules and algorithms. The batch modelling approach of PM minimizes user interaction and labour. The PM technique has been used with roof type information that is derived via DL to be able to construct roof geometries.

LOD1 block models have been generated via footprint extrusion [10, 11]. Roof geometries have been constructed using the Straight Skeleton Algorithm [12] and Sweep Line Algorithm [13]. Then, roof geometries are added to the top of block models. RESTful web services have been developed with web technologies to be able to implement the proposed methodology as a web-based solution. The entire computational cost of the model generation has been loaded to the server. Finally, constructed roof geometries and generated 3DCM have been visualized in the client's browser via HTML5 and WebGL.

There are two main novel contributions presented in this work. First, the Straight Skeleton algorithm has been extended using the Sweep Line algorithm to construct 3D roof geometries whose types are gathered with aerial imagery. Then, a web-based architecture for the generation of the LOD2 building models with condominium units has been proved and implemented.

The rest of the paper is organized as follows. In Section 2, the related studies to our study are criticized. In Section 3, the general architecture and the development essentials are described. In Section 4, experimental results and possible drawbacks are discussed. Finally, Section 5 states the conclusion and the future work.

## 2. Related Work

One widely used method to generate 3DCM is the extrusion from 2D footprints [10, 11]. Being the most discernable city object of the building footprints makes them the main subject of extrusion studies [14]. Nevertheless, 3DCMs generated by this method lack roof geometries. The lack of roof geometries hinders the widespread use of 3DCMs. While some 3D spatial analysis can be performed using LOD1 data, such as shadow analysis, others, such as solar potential, require minimum LOD2 data [5, 6].

Another popular method to generate 3DCM as LOD2 is using LIDAR point cloud data [15, 16, 17]. Roof shape and building shell is the most valuable geometric information that can be extracted from point cloud data. However, this process requires additional data such as 2D building footprints and computationally intensive data process workflows to be able to classify and construct city objects.

Recent studies have been done based on DL to obtain roof type information. In practical terms, deep learning is a subfield of machine learning (ML) and functions similarly, but it has different capabilities. Unlike other machine learning techniques, deep learning models can learn the discrimination between classes in a given dataset without feature extraction. On the other hand, deep learning techniques need more training data than ML techniques such as Support Vector Machine (SVM), Decision Trees, Logistic Regression, etc. because these ML techniques use only feature vectors extracted from raw data [18]. Deep learning techniques or more specifically CNNs are widely used for visual cognitive tasks as a classification to derive discriminant functions between classes from images. [19] used CNNs to classify roof types for multi-copter emergency landing site selection. First, they used the polygon of the building roof outline for cropping data from a LIDAR-based DSM image. Then, they used the polygon for cropping from the RGB image in the same way. After the data preparation, they fuse RGB and LIDAR images as input to CNN. [20] chose a deep learning-based approach. They used LIDAR-based DSM and pan-sharpened VHR (Very High Resolution) satellite images, but their approach has drawbacks with non-rectilinear roof shapes. [21] used CNNs with photogrammetric point clouds obtained from aerial images for classifying roof types. But they consider the roofs as only two classes: ridge and flat roofs. [5] have obtained roof types from LOD1 models with machine learning without using point clouds, but they have not yet constructed roof geometries using this information. Also, they used the LOD1 city model, not 2D building footprints.

## 3. Methodology

# 3.1. Obtaining Roof Type Information

A Python script was written to generate train images from aerial images. These images served as open data and can be found in [22]. Geopandas library was used to extract geometry and attributes from 2D footprints. With this script, multiple buildings were cropped from an aerial image and saved as separate image files using building envelopes that are extracted from 2D building footprints. Using Microsoft Custom Vision Service [23] that eases training process, training images were labeled as “Hipped”, “Gable”, “Flat” and “Negative” in train data (Fig. 1). The “Negative” label is used when the image cannot be identified as hipped, gable or flat. A general workflow for the training process is shown in Fig. 2.

After the training process, the model has reached 89,9% accuracy in predicting roof type. The trained model was exported from Custom Vision as a TensorFlow graph file to be used in a prediction process. A trained model can be saved as a model file and can be used in a different environment or application for the prediction process. This is often called “transfer of learning” in literature. A Python script was also coded to predict roof types. The model file was parsed using TensorFlow library [24] and roof types were predicted via this script automatically. A shapefile which consists of 2D building footprints were enriched with roof types as an attribute. All the workflow for prediction of roof type can be found on a publicly available Github repository that includes train data, test data and scripts.

# 3.2. Extrusion and Block Model Generation

A 3D model of the city can be obtained via extrusion of the building footprints. For this purpose, normal vectors of the building footprints are calculated. The normal vector is a unit vector perpendicular to the surface (Fig. 3). Extracting P1 from P2 and P2 from P3 vectors V1 and V2 are obtained. Using the cross product of these vectors, surface normal is calculated.

After the calculation of surface normals, using the “height” attribute of the building footprints, building footprints are translated along surface normals and top surfaces are constructed. By indexing the top and footprint points, vertical surfaces are constructed and LOD1 block model is derived. (Fig. 4).

There are condominium unit plans (CUP) drawn by geomatics engineers that have boundaries of condominiums as 2D (Fig. 5).

If CUPs are provided, our modeler can model condominium units as well at city scale. For this purpose, each condominium is translated along the surface normal according to the height of the condominium in CUP. Then, translated polygons are extruded and condominiums are modelled procedurally (Fig. 6).

Geometries of the condominium units are stored in a CityJSON file. Since condominium units are not defined in the CityGML data model, an ADE is developed for storing condominium units (Fig. 7).

## 3.3. Construction of Roof Geometries

A straight skeleton algorithm is used to model roof geometries. The definition of the straight skeleton (SS) is also its construction. SS is a geometric structure that consists of only straight edges, because of the stepwise shrinking process of a polygon (Fig. 8).

In this shrinking process, each edge of the polygon moves inwards of the polygon in a self-parallel manner. Two events change the topology of the input polygon. Edge Event occurs when an edge shrinks to zero and creates a node in skeleton Fig. 9 and Split Event occurs when a reflex vertex touches to a non-consecutive edge and creates a node in skeleton Fig. 10.

There are two approaches to stimulate this shrinking process. The 2D construction method is based on angular bisectors [26] and the 3D construction method is based on sweep planes firstly introduced by [27] and used by [28].

In the bisector-based approach, to create roof geometries in 3D, the skeleton must be traversed and Z values of nodes that are created by the skeleton must be manipulated according to proper roof height. Since sweep plane approach does not require this additional step, sweep plane approach has been used in this work.

In sweep plane approach, all nodes of the skeleton are detected by intersection of planes. Edge events have been detected by collision of three consecutive direction planes with the sweep plane and split events have been detected by collision of two consecutive and one non-consecutive direction planes with sweep plane.

In the proposed algorithm which takes polygons as input, vertices of polygons are stored in a circular doubly linked list, every vertex therefore, has pointers to consecutive vertices and also consecutive direction planes. Sweep plane, which is a plane parallel to the input polygon, moves in the direction of the Z axis and the polygon shrinks. Edge events and split events are stored in a priority queue according to the height of the sweep plane which gives the order of nodes in SS. When every edge of the input polygon shrinks to zero algorithm finishes and SS completed.

The result of SS is a directed graph. Roof surfaces must be generated from this graph to generate roofs. A 3D sweep line algorithm has been implemented to accomplish this. In this algorithm, first, nodes of SS are grouped according to the original edges of the input polygon. Then, for every group, a sweep line that is perpendicular to the related edge, moves from start to the end of the edge. Intersection order of nodes with this sweep line determines the order of the points for the roof polygon (Fig. 11). The horizontal roof surfaces are obtained using these orders. A predefined slope value is used to obtain the peak of the roof. Every roof surface is inclined from the edge to the skeleton node using the slope and roof geometry is obtained. In this work, the slope value is initially considered as %45 because it is Turkey's maximum allowed roof slope value (Fig. 12). However, this information could differ in various zoning regulations. This parameter could be set by the user or obtained from the attribution of 2D footprint data.

This extension of the SS algorithm directly gives the Hipped type roof geometry. It needs a modification if the subject is a Gable type of roof. The closest skeleton nodes to the start and end edges of the building footprint must be extended onto these edges. The slope value becomes 90% on these surfaces. If the subject is a Flat type of roof, there is no need to use the SS algorithm and the roof geometry is directly the same as the building footprint.

SS and sweep line have been implemented via web services as a web-based solution using RESTful architecture. The web application consists of 3 sub-modules: storage module, process module and web server module. The general system architecture is given below (Fig. 13). The application has been deployed on an Amazon S3 Bucket cloud storage environment. 2D building footprints can be uploaded to a MongoDB instance on the bucket via a rest service that has been developed using the Java Jersey web framework.

## 3.4. Visualization of the 3DCM via Browser

3DCMs are generally huge in size. Thus, clustering techniques should be used while passing them to client from a server [29]. Tiling approaches are used with web technologies such as HTML5 and WebGL to be able to visualize them via browsers. 3D roof geometries are added to 3D models of the tiles and with this, generated roofs are integrated to the tiling system. Thus, only needed tiles based on the user's current view in the scene are fetched from the server.

Since WebGL supports only triangles as primitives for representation of surfaces, 3D polygons must be triangulated. An “Ear Clipping” algorithm is used for this purpose and 3D polygon surfaces that belongs to roof geometries have been triangulated.

After generation of the tileset, to visualize the tileset in browser, open-source JavaScript library Cesium.js is used in this study. Since Cesium.js is built on WebGL, 3DCM has been rendered using the client's GPU and without any additional plug-in (Fig. 14).

## 4. Findings And Discussion

# 4.1. Performance Metrics of the Roof Type Prediction

3423 roof images that belong to hip (835), gable (1762) and flat (751) roofs were used in 0,3/0,7 of test/train ratio. The accuracy in predicting roof types is %89,9. Performance metrics are represented as AP (Average Precision), Recall and Precision (Fig. 15). The overall AP value of 0.89 proves the training is enough for prediction.

The proposed method has performed better than similar works in the related literature. The main reason for this result is that in this work, only 3 roof types have been classified while in [5], as the most similar work, 6 different roof types are classified. They have used the geometries of LOD1 3DCM to obtain roof types. The use of cognitive processes with aerial imagery increases overall performance compared to [5]. The accuracy of prediction for individual roof types is given as a confusion matrix (Table 1).

Table 1
Confusion matrix of prediction process

| | Flat | Gable | Hipped | Negative |
|---|---|---|---|---|
| Flat | 53 | 5 | 4 | 0 |
| Gable | 8 | 96 | 10 | 0 |
| Hipped | 1 | 14 | 143 | 0 |
| Negative | 1 | 10 | 2 | 14 |
| Total | 63 | 125 | 159 | 14 |

The comparison of predicted results with ground truth values could be found in Table 2. The precisions of the classes are not directly proportional with counts of images that are used to train the model. Because, as the quantity of the images affect the results, so does the quality of them.

Table 2
Performance metrics of prediction process

| Class | Ground Truth | Prediction | Accuracy | Precision | Recall |
|---|---|---|---|---|---|
| Flat | 63 | 62 | 94,74% | 0,85 | 0,84 |
| Gable | 125 | 114 | 86,98% | 0,84 | 0,77 |
| Hipped | 159 | 158 | 91,41% | 0,91 | 0,9 |
| Negative | 14 | 27 | 96,40% | 0,52 | 1 |

## 4.2. Special Cases and Floating-Point Issues

Implementing SS which requires high coordinate precision in a floating-point environment was caused some degenerations that the construction algorithm must handle. While calculating intersections between planes and determining new positions of vertices, some vertices that must have exact same coordinates can have different coordinates with minimal difference due to the floating-point arithmetic. A tolerance value must be predefined to handle this situation, and vertices with small coordinate differences with this tolerance value must be unionized into one vertex. If these situations are not handled, these slight differences can lead to tremendous changes in roof geometries (Fig. 16).

## 4.3. Limitation

Even if the web application can recognize the 3 most common roof types, it could be impossible to construct the roof properly when another roof type is the subject such as hexagonal, M-shaped, shed, or a combination of mentioned ones. The DL model classifies this different type of roof as Negative or one of the 3 main types which are the most similar to the roof and the roof construction process is done incorrectly. This construction of the roof geometries could cause incorrect results in analyses such as shadow, solar potential or visibility etc. Concerning this, the DL model’s recognizing ability could be improved by retraining the DL model with new roof types and image samples.

# 5. Conclusion

In this work, the generation of LOD2 building models with condominiums from 2D datasets without any user initiative is proved by the developed architecture. Roof geometries which are necessary for accurate solar potential or shadow analyses were constructed using the DL-generated roof type information and the novel SS extension. By the virtue of the CUPs, 3D models of condominiums could also be generated. Besides that, the entire pipeline of the model generation is executed in the server and the results are observable in the client device even if it's a low-cost device. Furthermore, all software development stages and tests were realized using open-source development tools.

During the preparation of this paper, the proposed pipeline for automatic modelling of roof geometries is limited to 3 classes: hip, gable and flat. Hence, there is ongoing work on this topic, and this work will be extended to include modelling other roof types such as the hexagonal, shed etc. Also, this work will aim to detect and model chimneys on the roofs as well. The detection and modelling of chimneys could be beneficial for the optimization of solar panel implementation in solar analyses. As another future work aim, the direct 3D reconstruction of the roof geometries from non-stereo aerial images will have experimented.

# Declarations

The authors declare no conflict of interest.

### Author Contribution

Ziya Usta and Alper Tunga Akın performed algorithmic development stages and prepared figures. Ziya Usta, Alper Tunga Akın and Çetin Cömert. wrote the main article text.

### Funding

No funding was obtained for this study.

### Code availability

https://github.com/alpertungakin/Roof-Type-Classification

# References

1. Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., Çöltekin, A.: Applications of 3d city models: State of the art review. ISPRS International Journal of Geo-Information 4(4), 2842–2889 (2015)
2. Welle Donker, F., Van Loenen, B., Bregt, A.K.: Open data and beyond. ISPRS International Journal of Geo-Information 5(4), 48 (2016)
3. Biljecki, F., Ledoux, H., Stoter, J.: Generating 3d city models without elevation data. Computers, Environment and Urban Systems 64, 1–18 (2017)
4. Biljecki, F., Ledoux, H., Stoter, J.: Does a finer level of detail of a 3d city model bring an improvement for estimating shadows? In: Advances in 3D Geoinformation, pp. 31–47 (2017)
5. Biljecki, F., Dehbi, Y.: Raise the roof: Towards generating LOD2 models without aerial surveys using machine learning. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 4, 27–34 (2019)
6. Weiler, V., Stave, J., Eicker, U.: Renewable energy generation scenarios using 3d urban modeling tools —methodology for heat pump and co-generation systems with case study application. Energies 12(3), 403 (2019)
7. Bengio, Y., Goodfellow, I., Courville, A.: Deep Learning vol. 1. MIT press Cambridge, MA, USA (2017)
8. Tsiliakou, E., Labropoulos, T., Dimopoulou, E.: Procedural modeling in 3d gis environment. International Journal of 3-D Information Modeling (IJ3DIM) 3(3), 17–34 (2014)

9. Martinovic, A.: Inverse procedural modeling of buildings (2015)
10. Ledoux, H., Meijers, M.: Topologically consistent 3d city models obtained by extrusion. International Journal of Geographical Information Science 25(4), 557–574 (2011)
11. Arroyo Ohori, K., Ledoux, H., Stoter, J.: A dimension-independent extrusion algorithm using generalised maps. International Journal of Geographical Information Science 29(7), 1166–1186 (2015)
12. Aichholzer, O., Aurenhammer, F., Alberts, D., G¨artner, B.: A novel type of skeleton for polygons. In: J. UCS The Journal of Universal Computer Science, pp. 752–761. (1996)
13. Souvaine, D.: Line segment intersection using a sweep line algorithm. Tufts University (2005)
14. Nurkarim, W., Wijayanto, A. W. Building footprint extraction and counting on very high-resolution satellite imagery using object detection deep learning framework. Earth Science Informatics, 16(1), 515-532. (2023)
15. Tomljenovic, I., H¨ofle, B., Tiede, D., Blaschke, T.: Building extraction from airborne laser scanning data: An analysis of the state of the art. Remote Sensing 7(4), 3826–3862 (2015)
16. Bauchet, J.-P., Lafarge, F.: City reconstruction from airborne lidar: a computational geometry approach. In: 3D GeoInfo 2019-14$^{th}$ Conference 3D GeoInfo (2019)
17. Dehbi, Y., Henn, A., Gröger, G., Stroh, V., Plümer, L. Robust and fast reconstruction of complex roofs with active sampling from 3D point clouds. Transactions in GIS, 25(1), 112-133 (2021)
18. Henn, A., Römer, C., Gröger, G., Plümer, L.: Automatic classification of building types in 3d city models. GeoInformatica 16(2), 281–306 (2012). https://doi.org/10.1007/s10707-011-0131-x
19. Castagno, J., Atkins, E.M.: Automatic classification of roof shapes for multicopter emergency landing site selection. In: 2018 Aviation Technology, Integration, and Operations Conference, p. 3977 (2018)
20. Partovi, T., Fraundorfer, F., Bahmanyar, R., Huang, H., Reinartz, P.: Automatic 3-d building model reconstruction from very high resolution stereo satellite imagery. Remote Sensing 11(14), 1660 (2019)
21. Axelsson, M., Soderman, U., Berg, A., Lithen, T.: Roof type classification using deep convolutional neural networks on low resolution photogrammetric point clouds from aerial imagery. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1293–1297 (2018)
22. Bradbury, K., Brigman, B., Collins, L., Johnson, T., Lin, S., Newell, R., Park, S., Suresh, S., Wiesner, H., Xi, Y.: Aerial imagery object identification dataset for building and road detection, and building height estimation. figshare (2016)
23. Salvaris, M., Dean, D., Tok, W.H.: Cognitive services and custom vision. In: Deep Learning with Azure, pp. 99–128 (2018)
24. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Man´e, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J.,

Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Vi´egas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org (2015). https://www.tensorflow.org/

25. Çağdaş V., Taşınmaz Değerlemesi Ders Planı, https://slideplayer.biz.tr/amp/3151467/ (2012)
26. Felkel, P., Obdrzalek, S.: Straight skeleton implementation. In: Proceedings of Spring Conference on Computer Graphics (1998). Citeseer
27. Eppstein, D., Erickson, J.: Raising roofs, crashing cycles, and playing pool: Applications of a data structure for finding pairwise interactions. Discrete & Computational Geometry 22(4), 569–592 (1999)
28. Kelly, T.: Unwritten procedural modeling with the straight skeleton. PhD thesis, University of Glasgow (2014)
29. Chen, J., Li, J., & Li, M. Progressive visualization of complex 3D models over the internet. Transactions in GIS, 20(6), 887-902 (2016)

# Figures

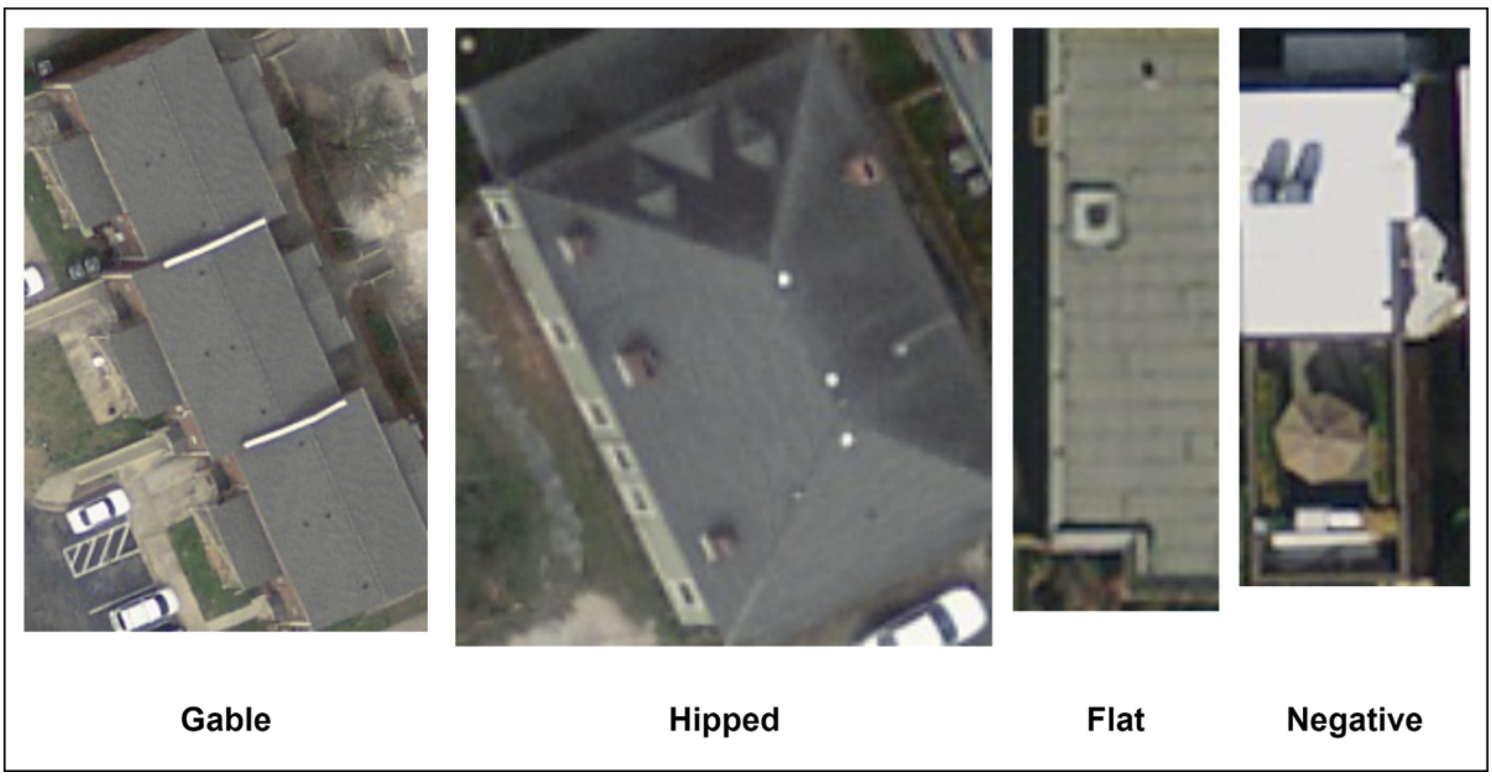
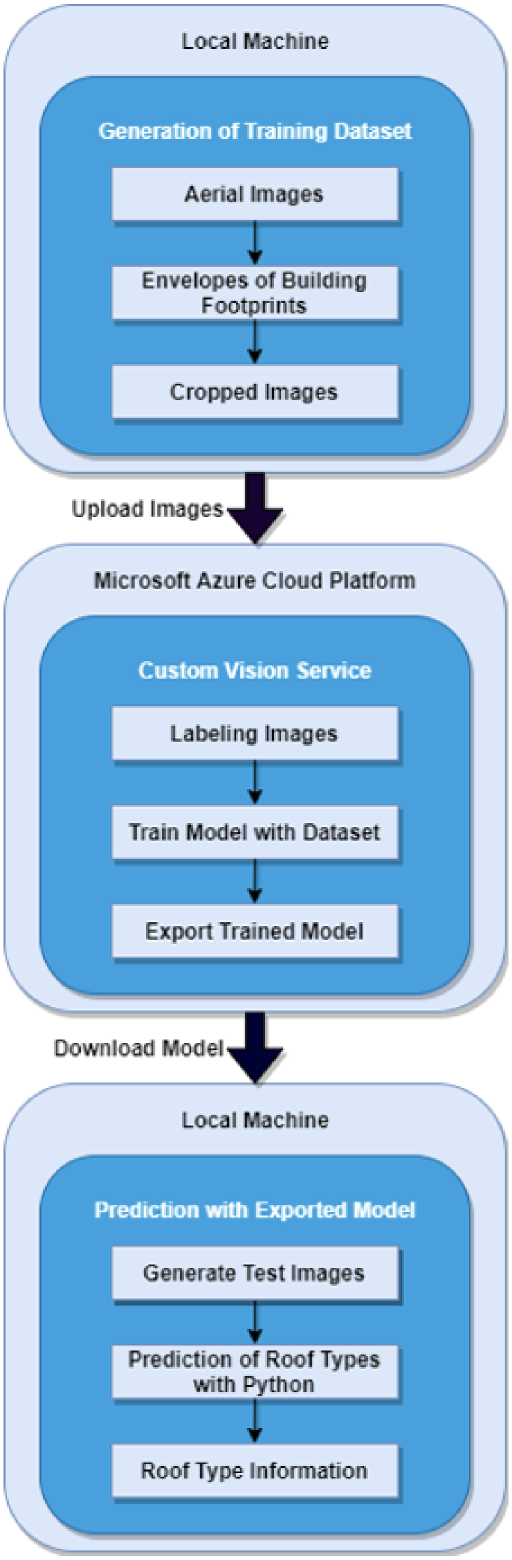



**Figure 1**

Samples from the roof type dataset.




## Figure 2

Training and Prediction Processes.

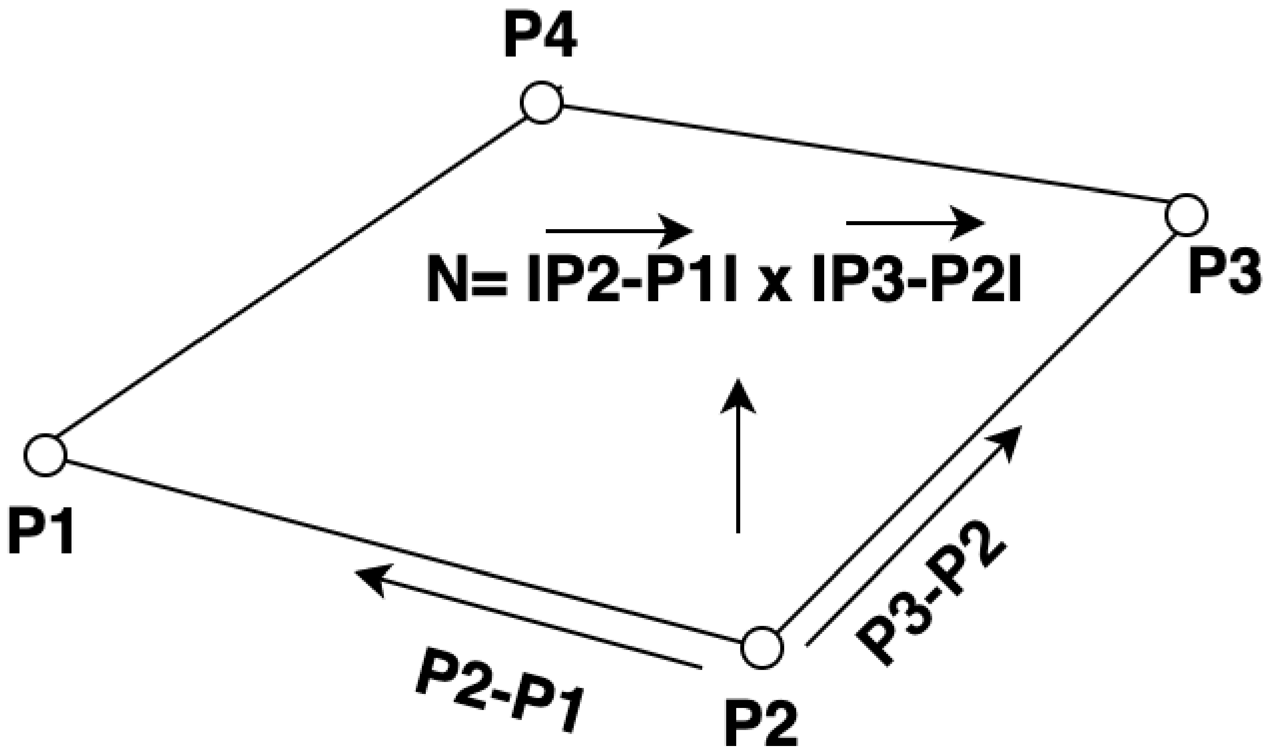



**Figure 3**

Calculation of the surface normal

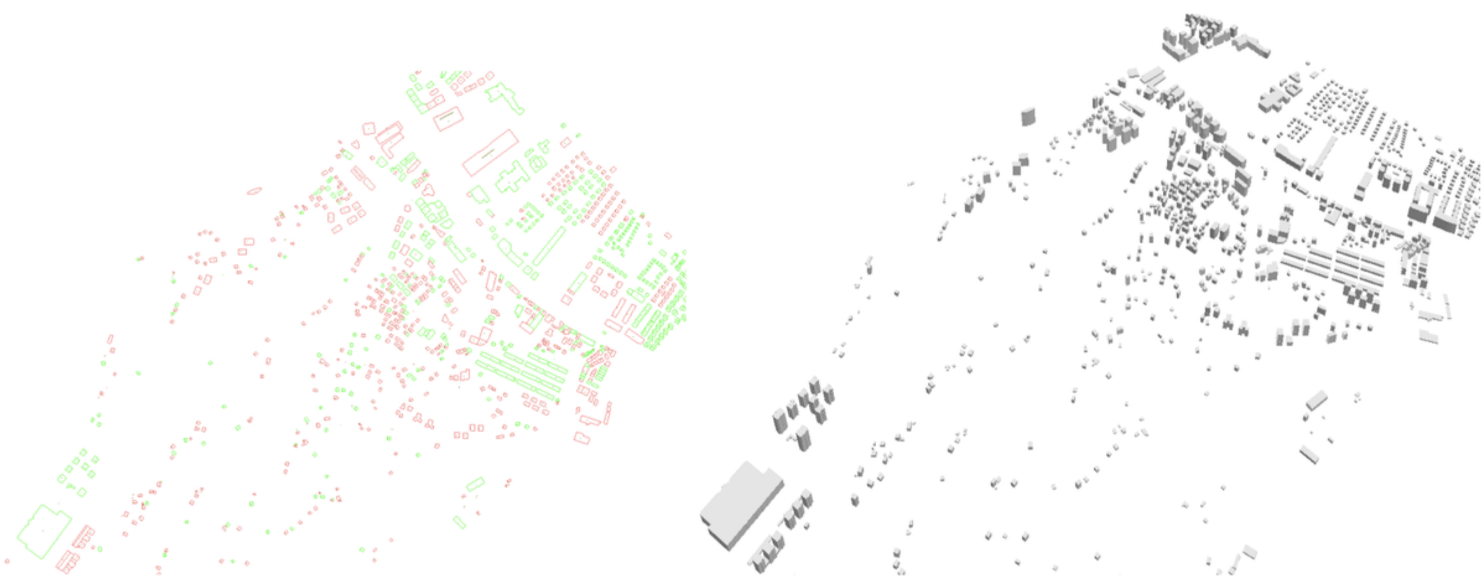
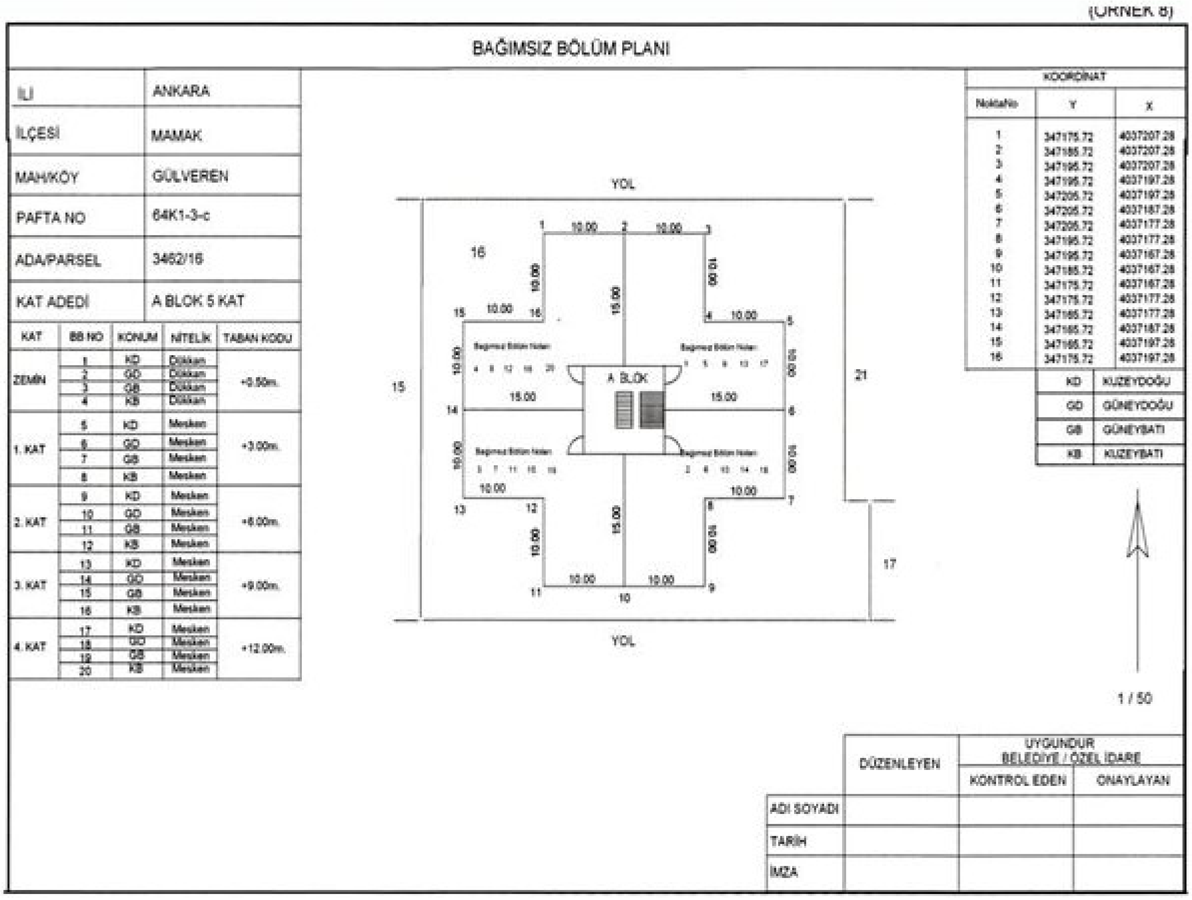


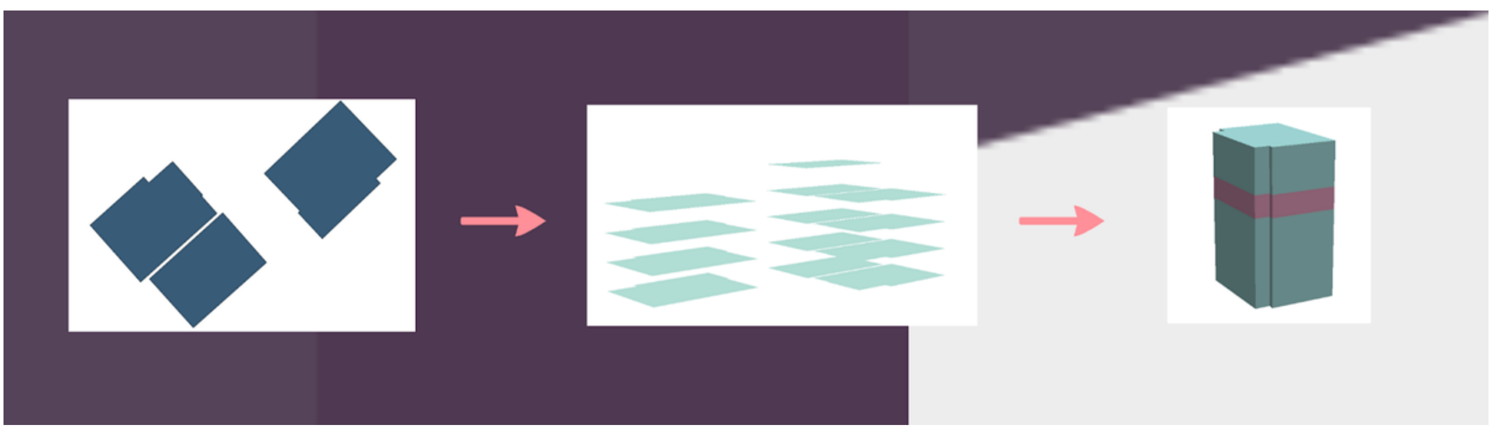
**Figure 4**

2D building footprints (left) 3D block model (right).




## Figure 5

Condominium unit plan [25]



## Figure 6

Polygons (left) multiplied and translated (middle) condominium unit in 3D.

```
{
    "type": "CityJSON",
    "version": "0.9",
    "metadata": {
        "geographicalExtent": [
            300060.181,
            5040722.785,
            8.444,
            301073.701,
            5042282.343,
            129.092
        ]
    },
    "CityObjects": {
  "B-206391182646-3B58D3D11C14": {
            "type": "Building",
            "condominiums": [
                {"id":"C-206391182646-3B58D3D11C14",
                    "type":"MultiSurface",
                    "boundaries":[[0,1,2],[0,2,3],[2,3,4]...]}],
            "geometry": [
                {
                    "type": "MultiSurface",
                    "boundaries": [
                        [
                            [
                                30203,
                                30204,
                                30205
                            ]
                        ],
                        [
                            [
                                30205,
                                30206,
                                30207
                            ]
                        ],
```


**Figure 7**

3D condominiums stored in the CityJSON file.

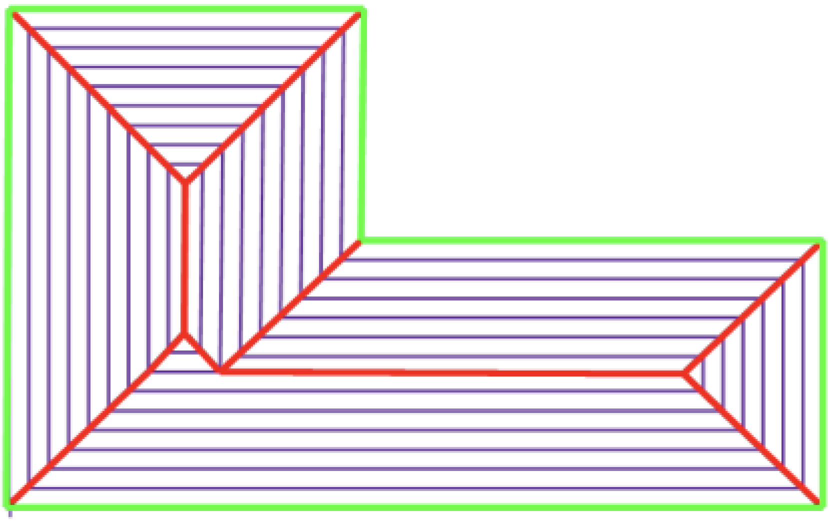


**Figure 8**

Input polygon (green), wave fronts (purple lines) and SS (red lines).

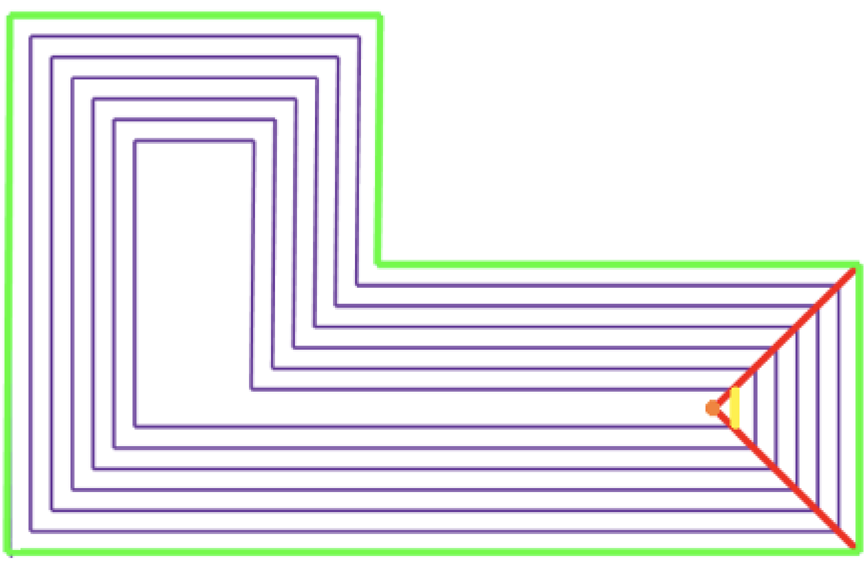


**Figure 9**

Edge Event (Yellow edge shrinks to zero at the orange point)

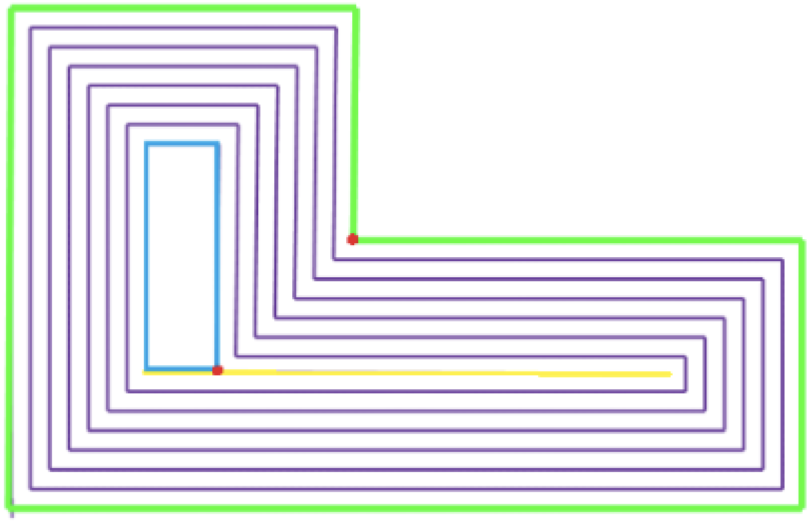
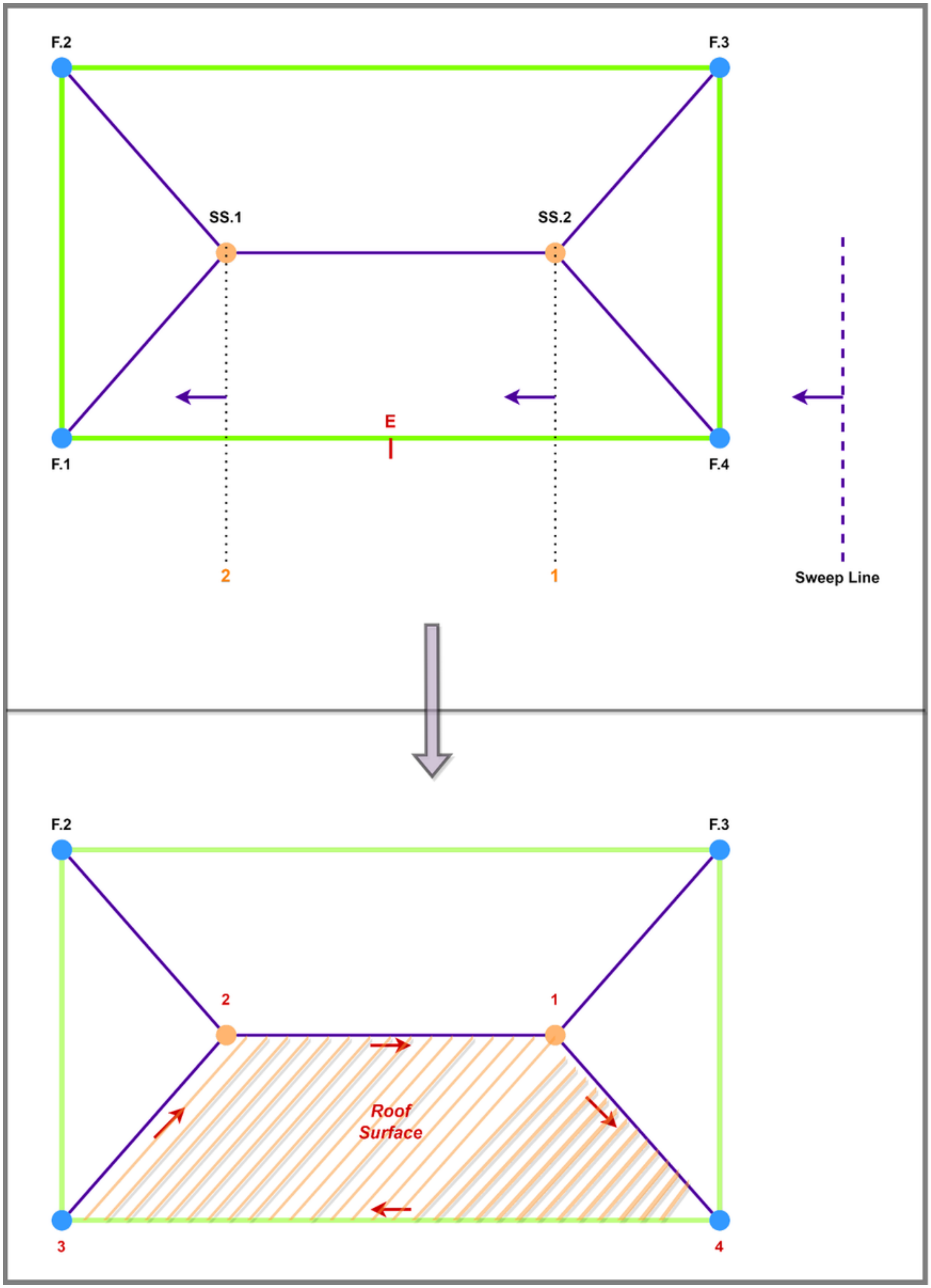


**Figure 10**

Split Event (yellow edge meets red reflex vertex and this creates blue polygon).




### Figure 11

Sweep Line Process. The orange nodes in the first row are belong to the SS. The purple dashed sweep line moves along the edge “E”. They are ordered according to the time of intersection with the sweep line. The roof surface is demonstrated with orange diagonal lines in the second row.

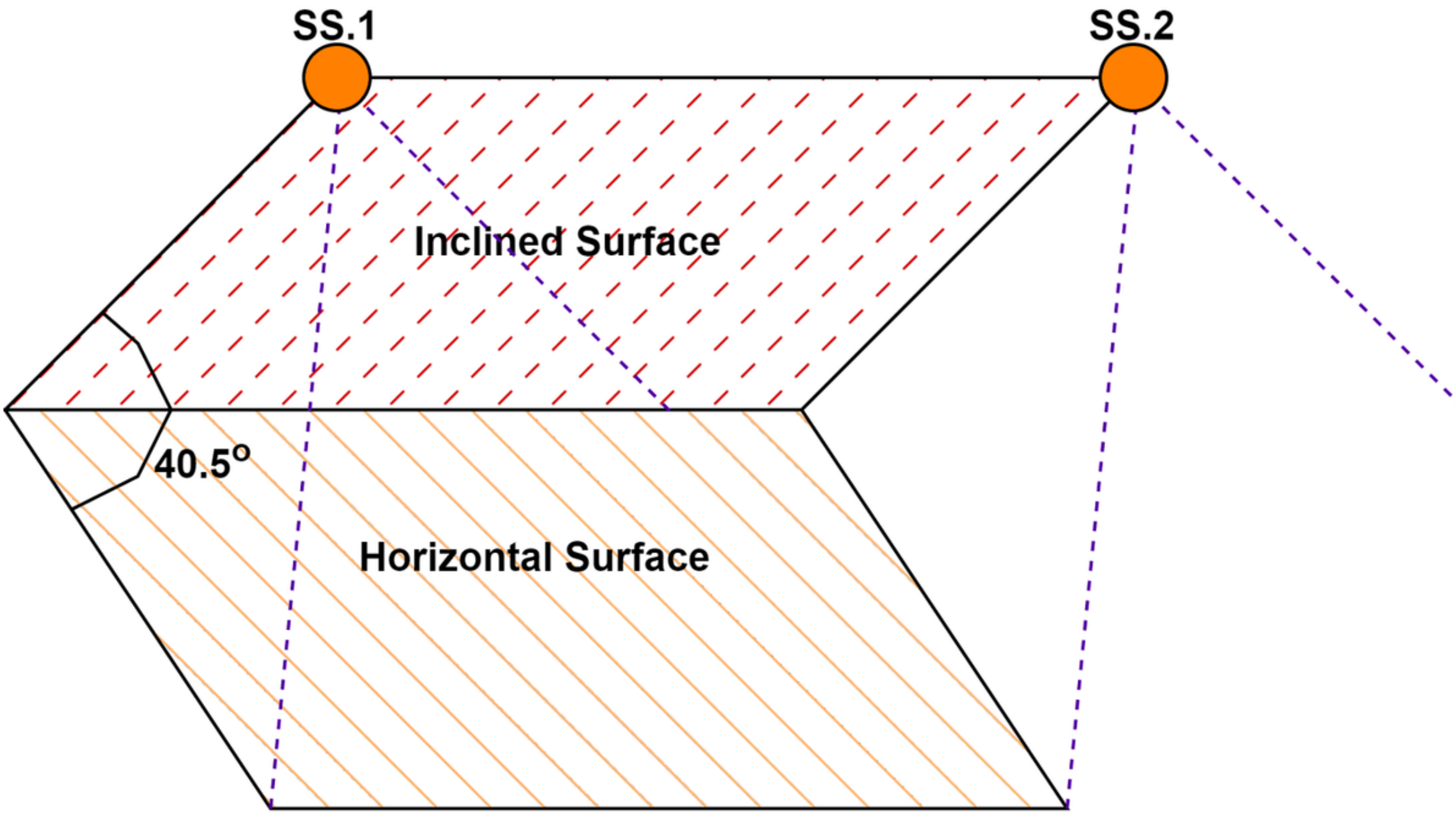



**Figure 12**

The incline process according to the slope value (45% = 40.5 degrees). The orange points demonstrate the SS nodes.

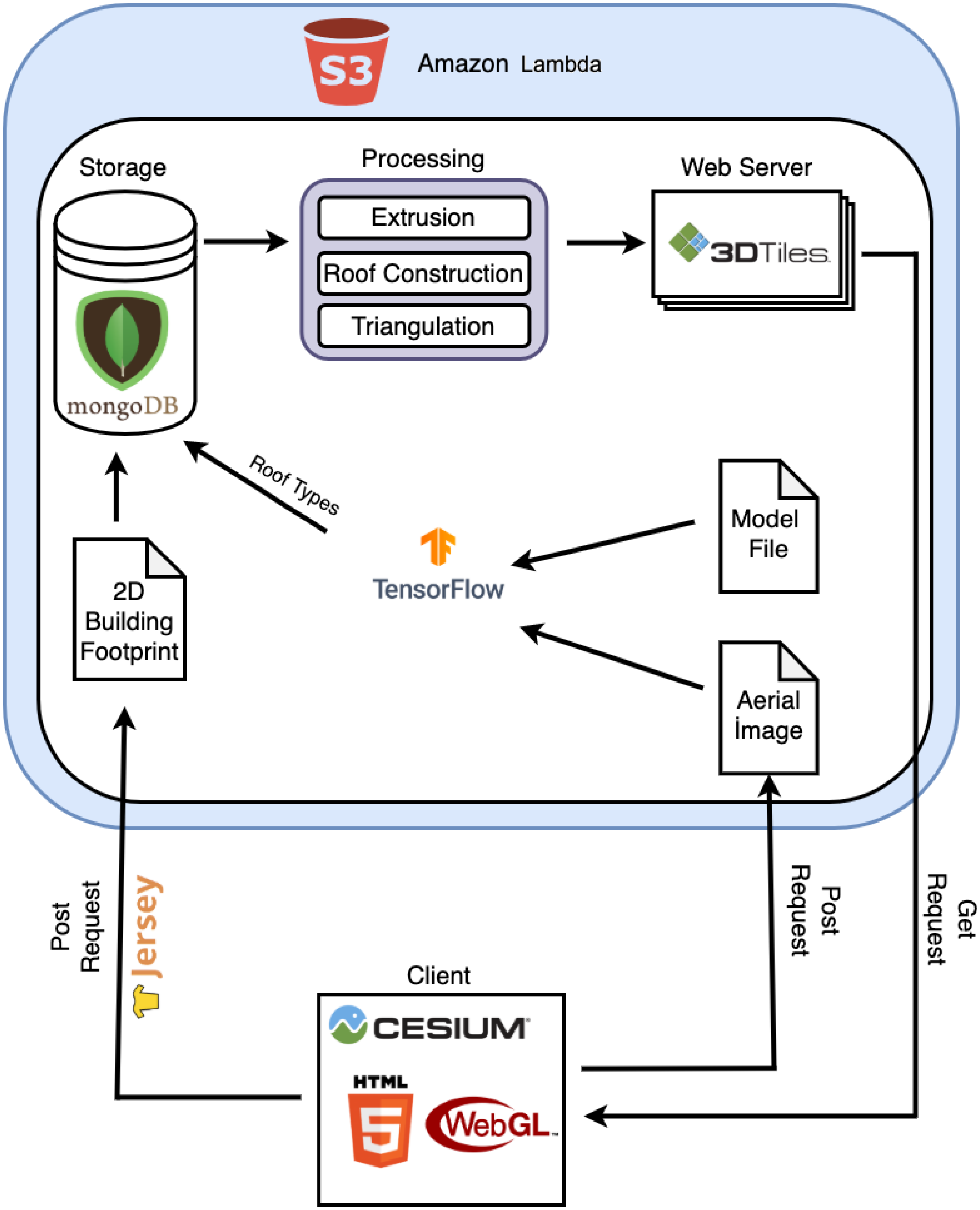



**Figure 13**

General System Architecture.

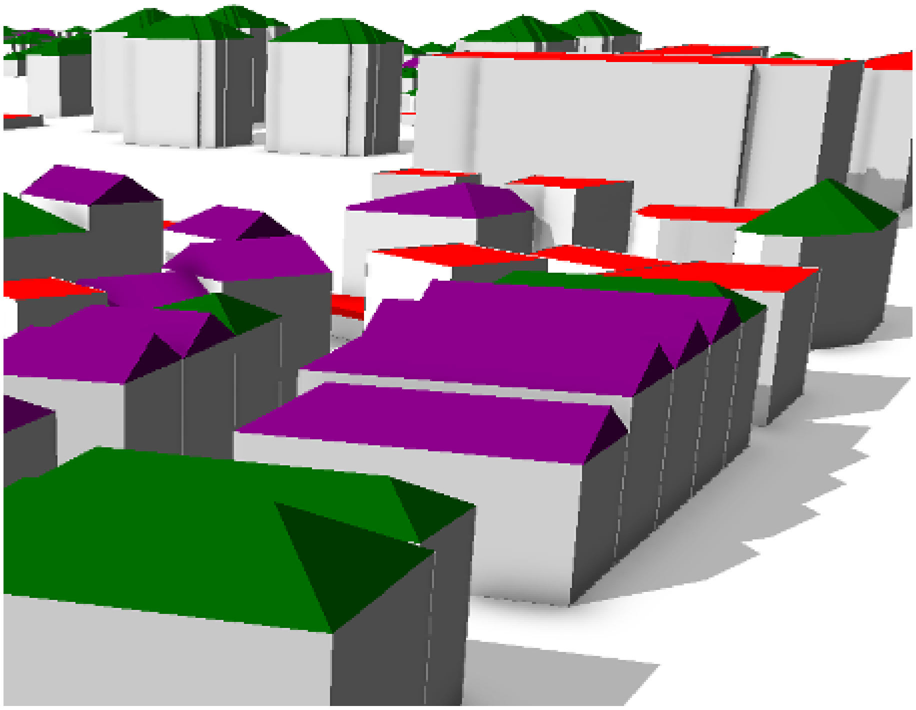


**Figure 14**

Red roofs: Flats, Green roofs: Hippeds, Purple roofs: Gables.

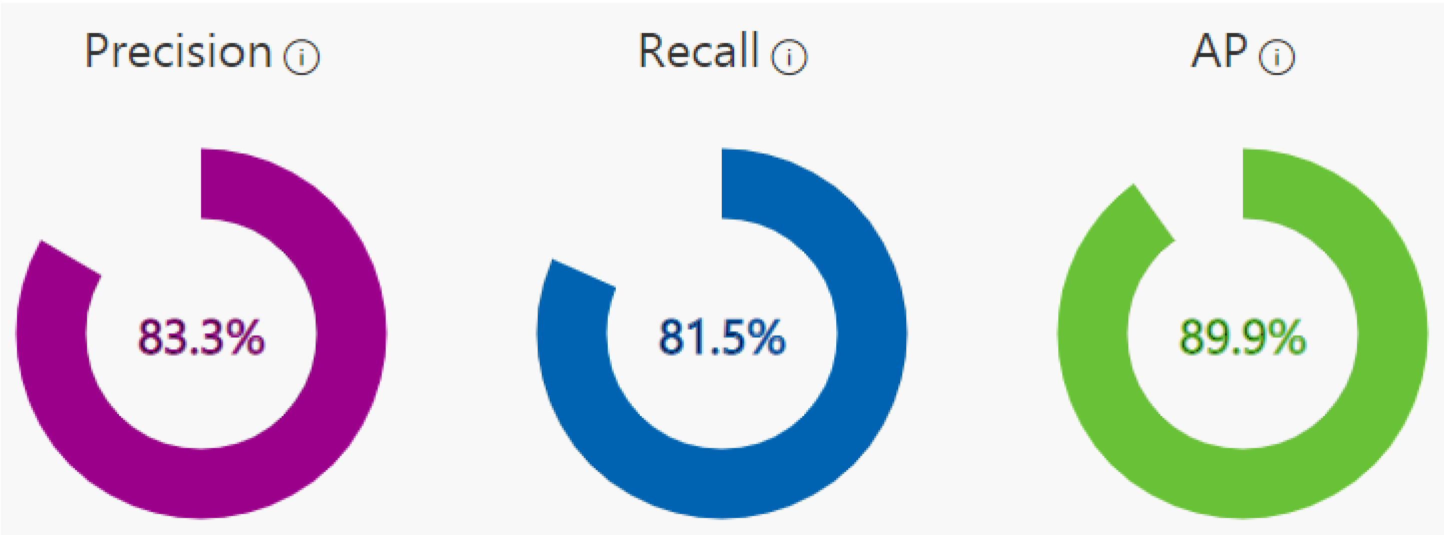
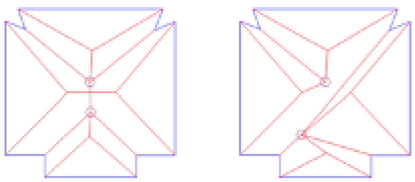



**Figure 15**

Performance metrics of training process.



## Figure 16

Degeneration. Two Edge Events must meet on the same point (left) but they do not properly meet (right).