Domain-Adaptive Object Detection for Enriching Semantic 3D City Models with Building Storeys from Street-View Images

Lukas Arzoumanidis, Al Maimun As Samee, Elmehdi Kanna, Son H. Nguyen, Youness Dehbi

Computational Methods Lab, HafenCity University, Hamburg, Germany - {firstname.lastname}@hcu-hamburg.de

Keywords: domain-adaptive learning, building storey estimation, object detection, semantic enrichment, 3D city models, CityGML.

Abstract

Semantically rich 3D city models play a vital role in a variety of applications, such as urban planning. Enhancing these models with currently unavailable attributes, such as building storey numbers, can unlock new opportunities to address pressing challenges, including sustainable urban development. In this work, we present an end-to-end pipeline for the automatic estimation of the number of storeys to semantically enrich 3D city models. We employ volunteered geographic information street-view imagery from Mapillary, using a COCO-pretrained object detection model to identify windows in façade images as key visual indicators for inferring building storey counts. Our detection pipeline, based on the YOLOv3 architecture, estimates storey numbers using an ensemble of clustering methods including Gaussian Mixtures and DBSCAN and enables the automatic augmentation of CityGML-based 3D city models by filling in missing attributes. This enrichment supports advanced applications, such as assessing building-scale energy demand, evaluating vertical urban growth patterns or population density estimations. We validated the feasibility of our approach with unfiltered Mapillary and applied it to a district in the city of Heidelberg, Germany. The paper also includes a detailed discussion of learning process quality, integration workflows, and visualization of the enriched 3D city model. The developed code is available at: `https://github.com/hcu-cml/citydb-buildingstoreys-ai`.

1. Introduction

As urban populations continue to expand, cities are compelled to adopt innovative, data-driven strategies to meet increasing societal demands while simultaneously promoting sustainable urban development. In this context, urban-analysis tasks, such as estimations on the population density, using low-cost sensors and open source geospatial datasets are essential to support urbanization monitoring and sustainable urban planning. Comprehensive semantic information about individual buildings, encompassing a wide range of physical and functional attributes, plays a pivotal role and further enables informed decision-making and supports evidence-based planning (Park et al., 2024; Sun et al., 2025; Arzoumanidis et al., 2025b).

Semantic information embedded within 3D building models, particularly details related to façade elements and structural attributes such as building height or number of storeys, offers invaluable insights for urban planners, especially when available at large-scale or city-wide resolutions. Although the value of such data is increasingly acknowledged, their cost-effective and efficient acquisition continues to pose a substantial difficulty. Reliable data on the number of building storeys are essential for identifying zones with potential for vertical densification. Leveraging this information facilitates the expansion of residential space without extensive demolition, thereby preserving the embodied carbon embedded in existing structures (Ding, 2018). Furthermore, the building height and number of storeys could provide a critical basis for estimating population capacity, as they are closely correlated with built-up density (Arzoumanidis et al., 2024). This relationship is particularly valuable in developing countries, where reliable information on population distribution is often scarce due to systemic mapping inequalities and limited data availability, including the absence of high-resolution point cloud or remote sensing datasets. Due to privacy concerns, even in developed urban areas, semantic information regarding the number of storeys is frequently not publicly accessible or openly shared, making this an important and continuing research topic across diverse urban contexts.

Accordingly, this work seeks to enhance semantic 3D city models by incorporating building storey information derived from a deep learning-based object detection and a complementary stochastic-geometric prediction module framework applied to volunteered geographic information (VGI) street-view imagery. A stochastic-geometric prediction module, combining Gaussian Mixture Models, k-means clustering, and DBSCAN (Ester et al., 1996) incorporates a majority-voting scheme to estimate the number of building storeys. The VGI street-view dataset is acquired from low-cost RGB sensors, such as smartphone cameras, providing six degrees of freedom and global positional information.

Our main contributions include:

- An end-to-end framework that enriches 3D city models by estimating building storey numbers from VGI street-view imagery,
- A domain-adaptive window-based façade analysis pipeline using a COCO-pretrained YOLOv3 object detector, fine-tuned on a normalized façade dataset,
- A stochastic-geometric prediction module combining Gaussian Mixture Models, k-means clustering, and DBSCAN with a majority-voting scheme for robust storey estimation,
- Demonstration and validation on real-world data, including integration into exsiting CityGML datasets for practical urban-analysis applications.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-4-2026-3-2026 | © Author(s) 2026. CC BY 4.0 License.

Figure 1. Visualization of buildings with enriched storey information in an excerpt of the test area of Heidelberg, Germany. For available Mapillary images, according building models are color-coded by number of floors based on our estimation: two (cyan), three (magenta), four (yellow), five (green), or none (gray). Visualized using CesiumJS and 3DCityDB Web Map Client, with ArcGIS World Imagery as the basemap.

Figure 1 depicts the results of our end-to-end approach. Based on our ensemble-based storey induction alongside a domain-adaptive window detection, the building models are enriched and highlighted using the 3DCityDB Web Map Client. The remainder of this paper is structured as follows: Section 2 reviews recent advances in the estimation of building height and storey numbers. Section 3 presents our proposed methodology in detail. Section 4 evaluates the quality and demonstrates the effectiveness of our approach. Finally, Section 5 concludes the paper and outlines directions for future research.

2. Related Work

Recent deep learning-based approaches for the prediction of building storeys typically rely on either single-source inputs, including street-view images (Iannelli and Dell’Acqua, 2017; Chen et al., 2022), satellite RGB imagery (Liasis and Stavrou, 2016; Liu et al., 2020), satellite radar data (Li et al., 2020; Kakooei and Baleghi, 2024) or on multi-source inputs, such as street-view imagery (Martínez Marín et al., 2023), OpenStreetMap (OSM) features (Li et al., 2023), 3D geospatial coordinates (Fan et al., 2024). Recently, a multi-source approach for storey estimation proposed by Li et al. (2023) composes three main components: semi-supervised learning from street-view imagery, extraction of multi-level morphometric features from OSM (e.g., building and street characteristics), and a building storey estimation framework utilizing a pre-trained façade object detection model. In a different work Tian et al. (2024) introduced an approach for large-scale building storey count estimation that relies on a normalized dataset generation pipeline deploying a Distillation with no labels (DINO) object detection model, alongside a multi-task deep neural network that leverages roof information to improve storey-number prediction accuracy.

In another multi-source approach, visual foundation models combined with building footprint data have been explored for height estimation using a convolutional neural network (CNN) augmented with attention mechanisms to extract high-level visual features, while a multilayer perceptron (MLP) is used to learn implicit building height information (Ge et al., 2025). The increasing availability of large-scale open-source Earth observation time-series datasets has further encouraged research on height prediction using temporal signals. One line of work employs regression models trained on highly accurate building height data derived from multiple 3D building models, in combination with Sentinel-1 and Sentinel-2 time series (Frantz et al., 2021). Another approach uses deep learning architectures explicitly designed to capture local and global spatiotemporal dependencies; for example, a T-SwinUNet model with a temporal attention module has been used to learn correlations between stable and dynamic building features across time to estimate building height (Yadav et al., 2025).

To improve the estimation of storey count and window-to-wall ratio from street-level imagery, Duran et al. (2025) proposed an approach that integrates multiple deep learning-based architectures in a single pipeline. A fully convolutional network (FCN) is employed within a transfer learning paradigm, leveraging a specialized façade dataset for domain-specific feature extraction and façade detection. The detected façades are subsequently cropped and provided as input to a SegFormer model with an encoder-decoder architecture. The encoder consists of a vision transformer, while the decoder is implemented as a MLP. This model is trained on a dedicated window dataset using a transfer learning strategy analogous to that applied in the façade detection stage. To identify candidate window regions and refine these regions, they developed a hybrid method combining Grounding DINO and SAM for the generation of high-

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-4-2026-3-2026 | © Author(s) 2026. CC BY 4.0 License.

resolution segmentation masks. For storey count estimation, the outputs of the window segmentation models are processed using DBSCAN clustering. The resulting number of clusters corresponds to the estimated number of building storeys.

Volunteered street-level imagery platforms, particularly Mapillary[1], have emerged as promising open alternatives to commercial services such as Google Street View. Unlike proprietary platforms, Mapillary operates under a free license and accepts user-contributed imagery collected with consumer devices. Nevertheless, SVI, such as Mapillary imagery, also presents challenges. Coverage can be spatially inconsistent (Fan et al., 2025), and the images are affected by occlusions, image distortions, varying viewpoints, and substantial visual noise (Hou and Biljecki, 2022; Biljecki and Ito, 2021).

3. Methodology

The following section describes our proposed end-to-end pipeline, which performs window detection on building façades, estimates building storey counts, matches the predicted building attributes with corresponding buildings from CityGML, and integrates the resulting information into a semantically rich CityGML model. In our work, we also propose a building storey estimation framework inspired by the ideas of Li et al. (2023). However, because the curated Mapillary dataset used in Li et al. (2023) is not publicly available, we were unable to compare our method directly against their baseline results. Moreover, we chose not to filter the heterogeneous and often challenging Mapillary imagery. Instead, our goal was to develop an approach capable of producing robust and reliable results across the full spectrum of available Mapillary images.

Domain-Adaptive Window Detection. Using the training dataset, we train the deep learning-based YOLOv3 model and YOLOv11n to detect windows in building façades, as illustrated in Figure 2. For our experiments, we adopted the Ultralytics[2] implementation of YOLOv3 and YOLOv11, which were pretrained on the COCO dataset prior to fine-tuning. The models were chosen for their strong generalization capability and widespread adoption, and their demonstrated suitability to detect tiny building rooftop objects (Arzoumanidis et al., 2025a) or building rooftop materials (Arzoumanidis et al., 2025b).

During training, model performance is assessed using two primary loss components: bounding box loss (box loss) and classification loss (cls loss). Classification loss measures the discrepancy between the predicted and true class labels, while bounding box loss quantifies the agreement between predicted and ground-truth bounding boxes by accounting for factors such as aspect ratio and the spatial offset between box centers. Additionally, we validate the performance of both models during training using Precision, Recall, mean Average Precision (mAP), and mAP_{50}, which are standard evaluation metrics in machine learning-based object detection tasks.

For post-training evaluation of our domain-adaptive window detection system, we employ Precision, Recall, and mAP as performance metrics. Computing mAP first requires determining the Intersection over Union (IoU) between predicted and ground-truth bounding boxes, along with the corresponding Precision and Recall values. IoU measures the ratio of the overlap area to the union area of the two boxes. Precision is defined as the proportion of true-positive detections among all predicted instances, whereas Recall denotes the proportion of true positives relative to all ground-truth instances. An IoU threshold is applied to ensure that only predictions with sufficient spatial correspondence are considered true positives. Precision-Recall (PR) curves are generated for each class by ranking predictions by confidence score and computing Precision and Recall across varying thresholds. Class-specific Average Precision (AP) values are then aggregated to derive the overall mAP. In this work, we additionally report mAP at an IoU threshold of 0.50 (mAP_{50}), corresponding to AP computed at IoU = 0.5.

Since taller structures often fall outside the camera's vertical field of view or are severely affected by perspective distortion in typical Mapillary imagery, our approach limits the estimation to a maximum of five storeys. All predictions exceeding five storeys are therefore binned into the five-storey class, as the imaging constraints prevent reliable estimation for taller buildings.

Stochastic-Geometric Estimation Module. The final storey-count estimation applies a majority-voting scheme across three clustering algorithms to enhance robustness against individual estimation failures. Specifically, we employ two geometric, unsupervised methods, K-means and DBSCAN, and one probabilistic method based on a Gaussian Mixture Model (GMM), as illustrated in Figure 2. The outputs of all three approaches are combined through majority voting to obtain a more stable and reliable storey estimate. A detailed analysis of these clustering strategies is beyond the scope of this work, as we aim to investigate this direction more thoroughly in future research.

3.1 Georeferencing and Matching Mapillary Images

To accurately assign predicted building storey numbers to their corresponding building footprints, we employ a spatial matching procedure. First, we extract from the original CityGML dataset of the same area the building identifiers, their bounding boxes, and, if available, their attribute *storeysAboveGround*. The extracted data is stored as a GeoJSON file using the ETRS89 / UTM Zone 32N (EPSG:25832). This GeoJSON file will later be extended with the predicted building storey numbers used for both evaluation and CityGML enrichment.

We then reproject these building footprints to WGS84 (EPSG:4326) to ensure compatibility with Mapillary's coordinate reference system. Afterwards, building centroids are computed from the projected polygon geometries and used as representative reference points for spatial matching. For each building, we create a 2D buffer with a radius of $200\,\text{m}$ around its centroid, and select all Mapillary images located within this buffer zone based on their camera position.

To exclude images that do not face the target building, we apply a bearing constraint. For each candidate image, the expected bearing θ_{expected} from the image position to the building centroid is computed using the forward azimuth formula:

$$\theta_{\text{expected}} = \left(\text{atan2}(X, Y) \cdot \frac{180}{\pi} + 360 \right) \bmod 360, \qquad (1)$$

where:

$$X = \sin(\Delta\lambda) \cdot \cos(\varphi_2),$$
$$Y = \cos(\varphi_1) \cdot \sin(\varphi_2) - \sin(\varphi_1) \cdot \cos(\varphi_2) \cdot \cos(\Delta\lambda).$$

[1] https://www.mapillary.com/
[2] https://docs.ultralytics.com/models/yolo11/

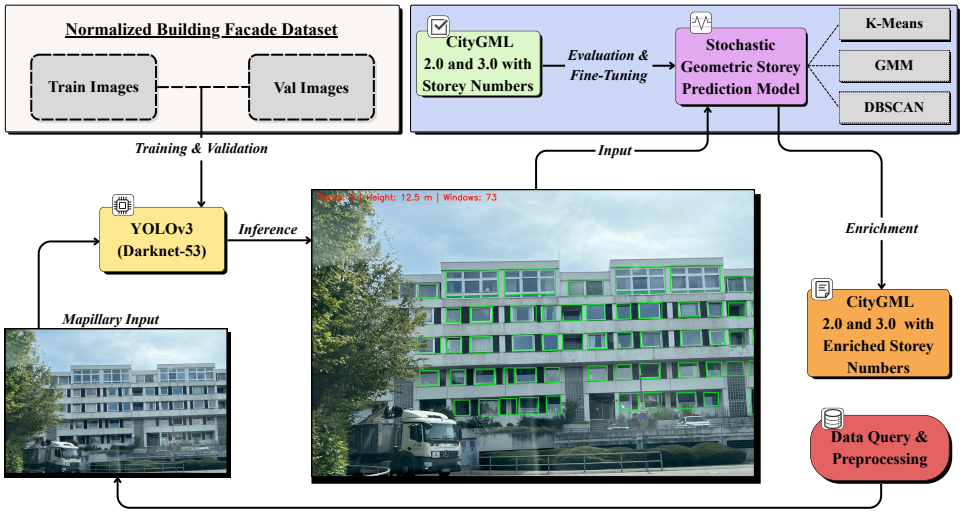


Figure 2. Overview of the proposed framework for storey prediction and semantic enrichment in 3D city models.

Here, (φ_1, λ_1) represents the image position, (φ_2, λ_2) denotes the building centroid position, $\Delta\lambda = \lambda_2 - \lambda_1$ (in radians), and θ_{expected} represents the compass direction from the image toward the building centroid. X and Y thus represent the directional components from the building centroid and image position.

The angular difference $\Delta\theta$ between the camera's compass bearing θ_{camera} and the expected bearing is then computed as:

$$\Delta\theta = \min\left(|\theta_{\text{camera}} - \theta_{\text{expected}}|,\ 360 - |\theta_{\text{camera}} - \theta_{\text{expected}}|\right) \tag{2}$$

Images with $\Delta\theta \leq 90^\circ$ are retained, ensuring that the building lies within the camera's approximate horizontal field of view, as illustrated in Figure 3.

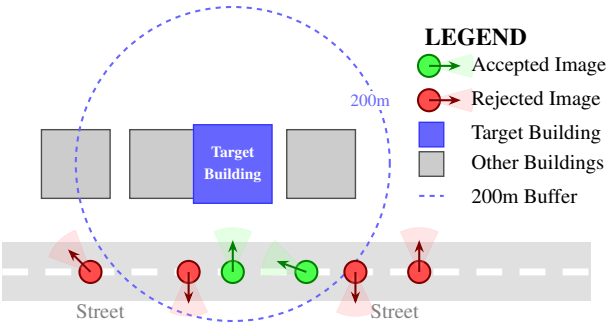


Figure 3. Schematics of matching Mapillary images to corresponding building footprints.

Each matched Mapillary image is then inferred using the YOLO model. The detection results (window counts and predicted number of storeys) are both embedded in the processed images and stored in the GeoJSON file introduced above as additional attributes associated with CityGML building identifiers.

3.2 Prediction Integration in Semantic 3D City Models

The City Geography Markup Language (CityGML) is an OGC-standardized information model and data exchange format designed to allow the representation and transfer of detailed 3D urban and landscape features (Gröger et al., 2012). In contrast to other 3D city modeling approaches that emphasize visual realism, such as those solely relying on aerial mesh reconstructions, CityGML couples geometric, topological, and appearance-based descriptions with extensive semantic information of city objects. The newest iteration of the standard, CityGML 3.0, was released by the OGC between 2021 and 2023 (Kolbe et al., 2021; Kutzner et al., 2023). We employ both versions 2.0 and 3.0 in our methods.

In real-world scenarios, applications rarely use general-purpose CityGML datasets in their text-based form. Instead, the data is often parsed and transformed further into application-specific models that allow for efficient storage and analysis. These representations may follow the graph-based (Nguyen, 2024) or relational data models (Yao et al., 2018).

Since the introduction of Structured Query Language (SQL), Relational Database Management Systems (RDBMSs) have become widely adopted across various fields (Davoudian et al., 2018). Many popular RDBMSs, including Oracle and PostgreSQL, offer built-in support for XML and GML data, and thus also allow for the storage of CityGML datasets given a predefined schema. Additionally, spatial extensions like PostGIS for PostgreSQL provide advanced tools for handling the important spatial information available in the CityGML data model. Among the most widely adopted solutions is the 3D City Database (3DCityDB) (Yao et al., 2018), an open-source, high-performance spatially-enhanced database designed for managing, visualizing, and exporting large CityGML datasets.

As of May 2025, the 3DCityDB has been released in its latest major version, 5.0, which fully supports CityGML 3.0 and features a completely redesigned, simplified database schema. These updates enhance both performance and flexibility compared to previous versions. Given the robustness of the 3DCityDB software suite[3] and the advanced capabilities of this new version, we selected version 5.0 and its accompanying utilities *citydb-tool*[4] to store and manage our CityGML test dataset.

The updated 3DCityDB schema *citydb* introduces a simplified structure, in which all feature types (such as buildings) are

[3] https://github.com/3dcitydb
[4] https://docs.3dcitydb.org/1.1/citydb-tool

stored in a single table *feature*, while most thematic attributes are managed through a unified table *property* (see Figure 4). In our use case, we utilize the attribute *storeysAboveGround* already defined in the CityGML data model to store the predicted numbers of above-ground storeys. This allows us to integrate our results directly into the model without introducing additional generic attributes or extending the database schema.

feature

id	objectid	envelope	...
1187	DEBW_5080004uy8W	01030000_	...
4012	DEBW_5080004uy4R	01030000_	...
71725	DEBW_0010002trxe	01030000_	...

Primary key

Foreign key

property

cid	feature_id	name	val_int	...
3255290	1187	storeys-	3	...
3255294	4012	Above-	4	...
3255416	71725	Ground	5	...

Figure 4. An excerpt of the enriched tables *feature* and *property* and their relationship for the Heidelberg dataset.

The integration of the predicted storey information into the 3DCityDB version 5.0 instance of the test CityGML dataset of Heidelberg is performed as follows:

1. For each building identifier in the CityGML dataset, retrieve the corresponding predicted number of storeys from the GeoJSON file produced in the previous step.
2. Using the building identifier, join the tables *feature* and *property* to determine whether the attribute *storeysAboveGround* is present in the table *property*:
   (a) If the attribute does not exist, insert the predicted number of storeys as a new property.
   (b) If the attribute does exist, update it only if the field *val_int* is empty; otherwise, keep the existing value.

To insert a new attribute *storeysAboveGround* into the database, we employ the following SQL statement:

```
INSERT INTO citydb.property (
    feature_id, datatype_id, namespace_id,
    name, val_int)
VALUES (id, 3, 10, 'storeysAboveGround',
   '2|3|4|5');
```

The values *datatype_id* and *namespace_id* are set to 3 and 10, respectively, for the attribute *storeysAboveGround* as defined in database schema of the 3DCityDB version 5.0.

The enriched city model can then be exported to both CityGML 2.0 and 3.0 using the 3DCityDB utilities *citydb-tool*.

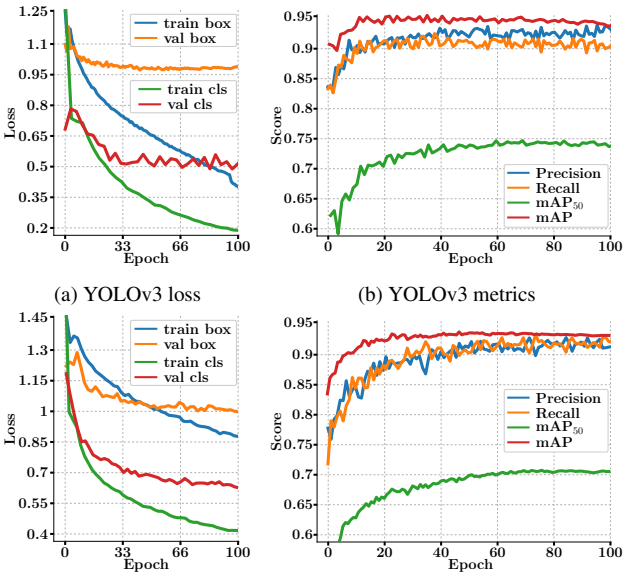
4. Experimental Results

The following section provides a detailed description of the standardized training dataset and presents the experimental results obtained using our approach, evaluated with commonly applied quantitative metrics for object detection.

Experimental Setup. To detect, classify, and map the different classes of roof material, we compared two pretrained deep learning-based object detection model, specifically YOLOv3 with a Darknet-53 backbone and YOLOv11n, as provided by Ultralytics. The models were pretrained on the COCO dataset[5], which comprises 80 object categories. Our implementation is based on PyTorch and utilizes CUDA acceleration. All experiments were performed using PyTorch version 2.2.0 with CUDA 12.8 on Google Colab using an NVIDIA L4 GPU. Default hyperparameter settings were used, and the model was trained for 100 epochs.

Training Dataset. This section details the preparation of the normalized dataset used to train our object detection model. Because the automated detection and mapping of windows must operate reliably across diverse building heights and façade configurations, we employ a large, heterogeneous, and normalized dataset for training, validation, and testing of the vision pipeline. For this work, we use the normalized window detection model dataset[6] from Roboflow (Roboflow, Inc., 2025) and convert it to the YOLO annotation format, enabling compatibility with the Ultralytics ecosystem of pretrained YOLO models.

4.1 Domain-Adaptive Window Detection



(a) YOLOv3 loss (b) YOLOv3 metrics

(c) YOLOv11 loss (d) YOLOv11 metrics

Figure 5. Training and validation loss, validation mAP, Precision, and Recall for YOLOv3 and YOLOv11.

The experimental results reveal substantial performance differences between the training, validation, and testing phases for both YOLO model versions.

During training, the loss curves for both models follow a consistently decreasing trend. As shown in Figure 5a and Figure 5c, the bounding-box regression loss and the classification loss decline within comparable ranges for the two architectures. This behavior is expected because both losses arise from related regression tasks: the bounding-box loss assesses spatial localization accuracy, whereas the classification loss reflects the correctness of predicted class labels. Since both

[5] https://cocodataset.org
[6] https://universe.roboflow.com/test-r8epu/object-detection-kifod

models are based on the same core principles, it is reasonable that the model learns to classify objects reliably before fully converging on optimal localization. Across all loss functions, the YOLOv3 model displays smoother loss trajectories and achieves lower overall loss values, as indicated by the differences in the y-axis scales in Figure 5a and Figure 5c. Evaluation of mAP, mAP_{50}, Precision, and Recall on the validation set shows monotonic improvement over epochs, as illustrated in Figure 5b and Figure 5d. The simultaneous decrease in training and validation losses, combined with rising validation metrics, suggests that both models learn effectively and generalize well. Notably, YOLOv3 converges substantially faster, reaching near-optimal performance by approximately Epoch 10, whereas YOLOv11 requires roughly 40 epochs to approach similar stability. Throughout training, YOLOv3 also maintains slightly superior validation performance across the evaluated metrics.

Transfer learning has become a fundamental paradigm in deep learning, enabling models pretrained on large, general-purpose datasets to be efficiently adapted to new tasks with limited annotated data (Pan and Yang, 2010; Yosinski et al., 2014). Rather than training a network from scratch, pretrained feature representations are reused and fine-tuned for a more specialized downstream application. This approach substantially reduces training time, improves generalization, and significantly decreases the volume of task-specific training data required.

Domain adaptation constitutes a more targeted form of transfer learning, addressing situations in which the source domain, used for training, and the target domain, used for deployment, differ in their data distributions (Wang and Deng, 2018; Ganin et al., 2016). Such differences, often referred to as domain shift, can markedly degrade performance when models trained on curated benchmark datasets are applied to real-world imagery, where lighting conditions, occlusions, and sensors vary widely. Domain-adaptive techniques mitigate these issues by aligning feature distributions across domains, employing strategies such as adversarial learning, feature-space, or regularization methods. These techniques allow models to preserve generalizable high-level features while adapting to the specific visual characteristics of the target environment.

In the context of window detection, domain adaptation is particularly important because publicly available façade datasets differ substantially from real-world Mapillary imagery in appearance, acquisition geometry, and overall quality.

4.2 Building Storey Estimation

Since our inference is conducted on real Mapillary images rather than on normalized façade imagery, we chose not to evaluate the normalized Roboflow test dataset. Instead, we assess the performance of the complete framework by evaluating the final building storey predictions derived from detected windows in original Mapillary images using our geometric-stochastic estimation module. To construct an appropriate test set, we collected a large number of suitable Mapillary images from Munich, Germany and matched them with 3D building models of the city. This approach is motivated by the fact that the CityGML dataset for Munich already provides authoritative information on the number of building storeys and also provides a great variety of building architectures, including taller buildings, such as, five storey buildings. As a result, we obtained a test dataset with reliable ground-truth storey counts, enabling an end-to-end evaluation of the proposed framework.

Table 1. Evaluation of per-storey Precision, Recall, and F1 scores on the generated ground-truth test dataset derived from Munich imagery.

storey	Precision	Recall	F1 score
1	0.38	0.50	0.43
2	0.17	0.31	0.22
3	0.33	0.44	0.38
4	0.49	0.48	0.49
5	0.60	0.43	0.50

For the evaluation of our approach, we selected the trained YOLOv3 model, as its validation results show a slight improvement over YOLOv11. As presented in Table 1, the F1 score indicates limited separability between two- and three-storey buildings, a pattern that is also apparent in the normalized confusion matrix in Figure 6. The model has difficulty distinguishing exact storey counts, with two-storey buildings being particularly challenging. However, accuracy improves substantially when a tolerance margin is allowed. The exact-match accuracy across all predicted storey numbers is 44.6%, with a mean absolute error of 0.8 storeys. Nonetheless, the accuracy within ± 1 storey reaches 72.1%, and the accuracy within ± 2 storeys increases to 94.6%, making the relatively low exact-match accuracy less critical than it appears at first glance.

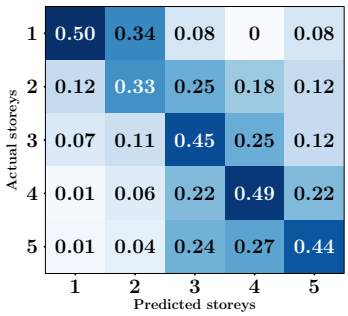


Figure 6. Normalized confusion matrix of the evaluated Munich dataset.

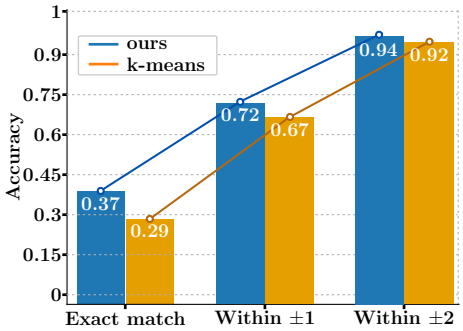


Figure 7. Comparison of our stochastic-geometric estimation model and the baseline K-means approach of Li et al. (2023) for image-footprint pairing accuracy, evaluated under exact-match conditions and with tolerances of ±1 and ±2 storeys.

This behavior can be attributed to the highly heterogeneous quality of Mapillary imagery. The images originate from a wide range of acquisition devices, including car dashcams (cf. Figure 8f), bicycle cameras (cf. Figure 8d), drones (cf. Figure 8b), and smartphones, and are captured from varying angles and lenses (cf. Figure 8a), elevations, and under diverse occlusion conditions, such as pedestrians (cf. Figure 8c) or parked

vehicles. Since no cherry-picking or candidate filtering is applied to the test images, these real-world imperfections are inherently reflected in the evaluation performance.

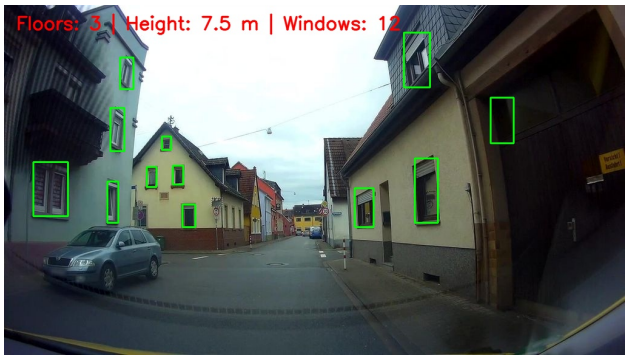
(a) ultra-wide-angle fisheye lens

(b) drone capture

(c) heavy occlusions

(d) successful prediction (left)

(e) successful prediction (front)



(f) successful prediction (left, right and front facing buildings)

Figure 8. Heterogeneity in Mapillary© images and associated storey number prediction based on image detection and our stochastic-geometric estimation module.

The stochastic-geometric estimation model shows a modest improvement in exact-match accuracy for image-footprint pairs compared with the baseline K-means approach proposed by Li et al. (2023), as illustrated in Figure 7. Furthermore, the same figure demonstrates that allowing a tolerance of one storey increases the accuracy by nearly 35%. When a mean absolute error tolerance of ± 2 storeys is permitted, the accuracy improves by an additional 20%, reaching values in the mid-90% range.

At first glance, these results may seem unimpressive. However, a closer examination reveals that the primary source of error originates from the georeferencing and matching pipeline used to associate images with building footprints, as well as from the often noisy location metadata of Mapillary images. Our approach depends on accurate GPS positions, which are frequently unreliable for low-cost consumer devices used for Mapillary images. This issue is particularly pronounced in street canyons with buildings taller than four storeys, as illustrated in Figure 9.

Compounding this problem, our approach generates and searches within a 200 m buffer for images that might capture the target building whose storey count is to be predicted. In dense urban environments, this relatively large search radius can result in images being incorrectly linked to adjacent buildings that fall within the buffer, specifically if the GPS is inaccurate. Consequently, the reference storey count may correspond to a different structure than the one actually depicted in the image and hence result in a false prediction in our evaluation metrics.

In future work, we aim to address the matching problem more rigorously by explicitly modelling the positional noise in the estimated camera locations. This appears to be a promising direction and will be the focus of subsequent research.

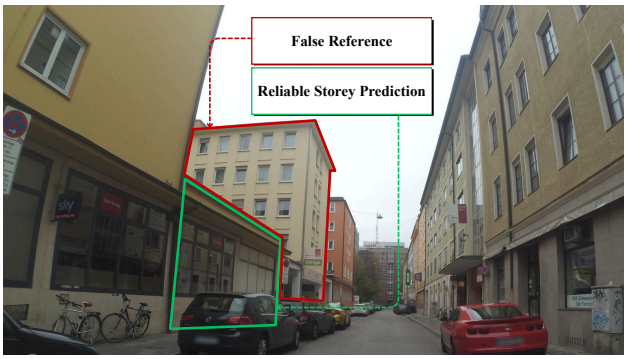


Figure 9. Mapillary© image illustrating a building where the model predicts a single storey correctly (green), while the ground truth contains five storeys (red).

4.3 Integration in CityGML

Since most buildings in the Munich CityGML dataset used for evaluating our approach already contain storey information, a dataset of another area is required in which the majority of buildings lack this attribute and for which a sufficient number of high-quality Mapillary images are available to support storey-number prediction. For this reason, we selected the CityGML dataset of the Heidelberg city center in the state of Baden-Württemberg, Germany. The study area covers approximately $8 \times 14\ \mathrm{km}^2$ (see Figure 10) and comprises 1.3 GB of data. None of the 67,222 buildings in this dataset have storey information, making it well suited for our prediction and enrichment task.

Due to the difference in Mapillary image quality between cities, for Heidelberg, we only consider buildings with at least eight detected windows. As a result, all single-storey buildings were excluded from the enriched dataset. After completing the prediction and enrichment process, a total of 4075 buildings were assigned new storey numbers ranging from two to five storeys. Of these, 1427 buildings (35 %) were predicted to have two storeys, 1222 (30 %) to have three storeys, 989 (24 %) to have four storeys, and 437 (11 %) to have five storeys. Their distribution is shown in Figure 11. Figure 1 shows a color-coded 3D visualization of these buildings within a representative excerpt of the study area.

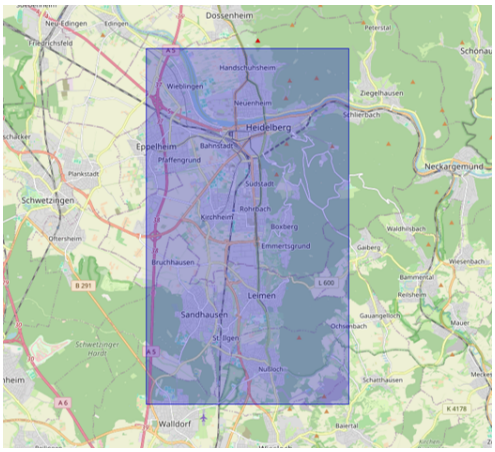


Figure 10. Bounding box of the test area of Heidelberg, Germany. Basemap from OSM©.

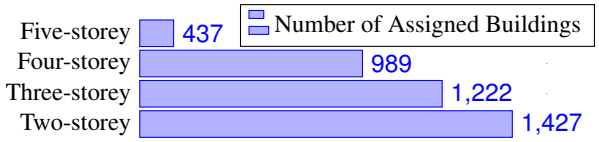


Figure 11. Distribution of buildings by estimated storey numbers in the Heidelberg dataset (without single-storey buildings).

5. Conclusion & Outlook

This paper presents an end-to-end pipeline for enriching semantic 3D city models with building-storey information derived from volunteered geographic information (VGI) street-view imagery. By combining a COCO-pretrained object detector with a domain-adaptive training strategy, the proposed method reliably detects façade windows under highly heterogeneous imaging conditions. The subsequent stochastic-geometric estimation module-integrating Gaussian Mixtures, K-means clustering, and DBSCAN within a majority-voting scheme-enables the inference of building storey numbers. The integration of these predictions into CityGML 2.0 and CityGML 3.0 using the 3DCityDB demonstrates the feasibility of large-scale semantic enrichment workflows that rely exclusively on openly available data and open-source tools.

Our experiments show that the method achieves robust performance in realistic conditions, despite substantial variability in image quality, sensor types, acquisition geometry, and occlusions. While exact storey classification remains challenging for some classes, particularly two-storey buildings, the high accuracy achieved within ± 1 or ± 2 storeys highlights the practical usefulness of the approach for a wide range of urban-analysis tasks. As VGI and open-data repositories continue to grow, such methods will play an increasingly important role in supporting urban analytics.

The semantic enrichment of the CityGML Heidelberg dataset further demonstrates how building-storey predictions can be integrated directly into established 3D city-model infrastructures. From a semantic-modelling perspective, the methodology is generalizable beyond storey estimation. Nonetheless, several limitations must be acknowledged. First, the practical upper bound of five storeys, imposed by the vertical field of view and perspective distortion common in Mapillary imagery, precludes reliable inference for taller buildings. Second, although the ensemble-based clustering strategy improves robustness compared with earlier approaches such as Li et al. (2023), variations in window design, façade irregularities, and architectural styles continue to hinder accurate storey estimation, even with domain-adaptive object detection on a normalized façade dataset.

Future work may explore façade-element segmentation and building-height prediction in areas lacking authoritative 3D datasets. In particular, the workflow could support the generation of LoD1 building models from OSM footprints using only VGI street-view data, providing a low-cost pathway for 3D reconstruction in cities without existing CityGML resources. Additionally, incorporating geometric cues such as vanishing lines, depth proxies, or monocular height-estimation models may help mitigate the challenges posed by missing or irregular façade features.

6. Acknowledgements

This research was partially supported by the project ‘Next Generation City Networking’ (Grant No. 19DZ24004) at the Hanseatic Wireless Innovation Competence Center (HAWICC). The project is funded by the Federal Ministry of Transport via the German Center for Future Mobility (DZM).

References

Arzoumanidis, L., Hecht, J., Dehbi, Y., 2024. Towards a Deep Automatic Generation of Figure-ground Maps. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W5-2024, 33–39.

Arzoumanidis, L., Li, W., Knechtel, J., Kosmayadi, Y., Dehbi, Y., 2025a. Automatic Detection of Tiny Drainage Outlets and Ventilations on Flat Rooftops from Aerial Imagery. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 125–132.

Arzoumanidis, L., Nguyen, S. H., Johannsen, L., Rothaut, F., Li, W., Dehbi, Y., 2025b. Object Detection for the Enrichment of Semantic 3D City Models with Roofing Materials. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W6-2025, 9–16.

Biljecki, F., Ito, K., 2021. Street View Imagery in Urban Analytics and GIS: A Review. *Landscape and Urban Planning*, 215, 104217.

Chen, F.-C., Subedi, A., Jahanshahi, M. R., Johnson, D. R., Delp, E. J., 2022. Deep Learning–Based Building Attribute Estimation from Google Street View Images for Flood Risk Assessment Using Feature Fusion and Task Relation Encoding. *Journal of Computing in Civil Engineering*, 36(6), 04022031.

Davoudian, A., Chen, L., Liu, M., 2018. *A Survey on NoSQL Stores*. 51Number 2, Association for Computing Machinery (ACM), 1–43.

Ding, G. K. C., 2018. *Embodied Carbon in Construction, Maintenance and Demolition in Buildings*. Springer International Publishing, Cham, 217–245.

Duran, A., Karapiperis, P., Waibel, C., Schlueter, A., 2025. Deep Learning-Based WWR and Floor Count Extraction from Façade Images to Improve UBEM. *Journal of Physics: Conference Series*, 3140(6), 062002.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X. et al., 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. 96(34), 226–231.

Fan, K., Lin, A., Wu, H., Xu, Z., 2024. Pano2Geo: An Efficient and Robust Building Height Estimation Model Using Street-View Panoramas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 215, 177–191.

Fan, Z., Feng, C.-C., Biljecki, F., 2025. Coverage and Bias of Street View Imagery in Mapping the Urban Environment. *Computers, Environment and Urban Systems*, 117, 102253.

Frantz, D., Schug, F., Okujeni, A., Navacchi, C., Wagner, W., van der Linden, S., Hostert, P., 2021. National-Scale Mapping of Building Height Using Sentinel-1 and Sentinel-2 Time Series. *Remote Sensing of Environment*, 252, 112128.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V., 2016. Domain-Adversarial Training of Neural Networks. *J. Mach. Learn. Res.*, 17(1), 2096–2030.

Ge, S., Liu, J., Che, X., Wang, Y., Huang, H., 2025. A Novel Method for Estimating Building Height from Baidu Panoramic Street View Images. *ISPRS International Journal of Geo-Information*, 14(8), 297.

Gröger, G., Kolbe, T. H., Nagel, C., Häfele, K.-H., 2012. *OGC City Geography Markup Language (CityGML) Encoding Standard*. Open Geospatial Consortium.

Hou, Y., Biljecki, F., 2022. A Comprehensive Framework for Evaluating the Quality of Street View Imagery. *International Journal of Applied Earth Observation and Geoinformation*, 115, 103094.

Iannelli, G. C., Dell’Acqua, F., 2017. Extensive Exposure Mapping in Urban Areas through Deep Analysis of Street-Level Pictures for Floor Count Determination. *Urban Science*, 1(2).

Kakooei, M., Baleghi, Y., 2024. Mapping Building Heights at Large Scales Using Sentinel-1 SAR Data and Deep Learning Methods. *Remote Sensing*, 16(18), 3371.

Kolbe, T. H., Kutzner, T., Smyth, C. S., Nagel, C., Roensdorf, C., Heazel, C., 2021. *OGC City Geography Markup Language (CityGML) Version 3.0 Part 1: Conceptual Model Standard*. Open Geospatial Consortium. International Standard.

Kutzner, T., Smyth, C., Nagel, C., Coors, V., Vinasco-Alvarez, D., Ishimaru, N., Yao, Z., Heazel, C., Kolbe, T. H., 2023. *OGC City Geography Markup Language (CityGML) Version 3.0 Part 2: GML Encoding Standard*. Open Geospatial Consortium. International Standard.

Li, H., Yuan, Z., Dax, G., Kong, G., Fan, H., Zipf, A., Werner, M., 2023. Semi-Supervised Learning from Street-View Images and OpenStreetMap for Automatic Building Height Estimation. *GIScience 2023, Leibniz International Proceedings in Informatics (LIPIcs)*.

Li, X., Zhou, Y., Gong, P., Seto, K. C., Clinton, N., 2020. Developing a Method to Estimate Building Height from Sentinel-1 Data. *Remote Sensing of Environment*, 240, 111705.

Liasis, G., Stavrou, S., 2016. Satellite Images Analysis for Shadow Detection and Building Height Estimation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 119, 437–450.

Liu, C.-J., Krylov, V. A., Kane, P., Kavanagh, G., Dahyot, R., 2020. IM2ELEVATION: Building Height Estimation from Single-View Aerial Imagery. *Remote Sensing*, 12(17), 2719.

Martínez Marín, R., González-Rodrigo, B., Marchamalo-Sacristán, M., 2023. Automatic Building Height Estimation: Machine Learning Models for Urban Image Analysis. *Applied Sciences*, 13(8), 5037.

Nguyen, S. H., 2024. Automatic Detection and Interpretation of Changes in Massive Semantic 3D City Models. PhD thesis, Technical University of Munich.

Pan, S. J., Yang, Q., 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345-1359.

Park, J., Park, S., Kang, J., 2024. Detecting and Classifying Rooftops with a CNN-based Remote-sensing Method for Urban Area Cool Roof Application. *Energy Reports*, 11, 2516-2525.

Roboflow, Inc., 2025. Roboflow public datasets. `https://public.roboflow.com/`. Accessed: 2025-11-15.

Sun, K., Li, Q., Liu, Q., Song, J., Dai, M., Qian, X., Gummidi, S. R. B., Yu, B., Creutzig, F., Liu, G., 2025. Urban Fabric Decoded: High-precision Building Material Identification via Deep Learning and Remote Sensing. *Environmental Science and Ecotechnology*, 24, 100538.

Tian, Y., Sun, Y., Zhu, X. X., 2024. Learning Building Floor Numbers from Crowdsourced Streetview Images. *Abstracts of the ICA*, 7, 171.

Wang, M., Deng, W., 2018. Deep Visual Domain Adaptation: A Survey. *Neurocomputing*, 312, 135–153.

Yadav, R., Nascetti, A., Ban, Y., 2025. How High Are We? Large-Scale Building Height Estimation at 10m Using Sentinel-1 SAR and Sentinel-2 MSI Time Series. *Remote Sensing of Environment*, 318, 114556.

Yao, Z., Nagel, C., Kunde, F., Hudra, G., Willkomm, P., Donaubauer, A., Adolphi, T., Kolbe, T. H., 2018. 3DCityDB - A 3D Geodatabase Solution for the Management, Analysis, and Visualization of Semantic 3D City Models based on CityGML. *Open Geospatial Data, Software and Standards*, 3(5), 1-26.

Yosinski, J., Clune, J., Bengio, Y., Lipson, H., 2014. How Transferable Are Features in Deep Neural Networks? *Advances in Neural Information Processing Systems (NeurIPS)*, 3320–3328.