



Article

End-to-End Framework for the Automatic Matching of Omnidirectional Street Images and Building Data and the Creation of 3D Building Models

Yoshiki Ogawa ^{1,*} , Ryoto Nakamura ², Go Sato ², Hiroya Maeda ³ and Yoshihide Sekimoto ¹

¹ Center for Spatial Information Science, The University of Tokyo, Tokyo 153-8505, Japan; sekimoto@csis.u-tokyo.ac.jp

² Department of Civil Engineering, The University of Tokyo, Tokyo 153-8505, Japan

³ Institute of Industrial Science, The University of Tokyo, Tokyo 153-8505, Japan

* Correspondence: ogawa@csis.u-tokyo.ac.jp; Tel.: +81-3-5452-6406

Abstract: For accurate urban planning, three-dimensional (3D) building models with a high level of detail (LOD) must be developed. However, most large-scale 3D building models are limited to a low LOD of 1–2, as the creation of higher LOD models requires the modeling of detailed building elements such as walls, windows, doors, and roof shapes. This process is currently not automated and is performed manually. In this study, an end-to-end framework for the creation of 3D building models was proposed by integrating multi-source data such as omnidirectional images, building footprints, and aerial photographs. These different data sources were matched with the building ID considering their spatial location. The building element information related to the exterior of the building was extracted, and detailed LOD3 3D building models were created. Experiments were conducted using data from Kobe, Japan, yielding a high accuracy for the intermediate processes, such as an 86.9% accuracy in building matching, an 88.3% pixel-based accuracy in the building element extraction, and an 89.7% accuracy in the roof type classification. Eighty-one LOD3 3D building models were created in 8 h, demonstrating that our method can create 3D building models that adequately represent the exterior information of actual buildings.

Keywords: 3D building model; building matching; omnidirectional image; building footprint; building element extraction; texture synthesis



Citation: Ogawa, Y.; Nakamura, R.; Sato, G.; Maeda, H.; Sekimoto, Y. End-to-End Framework for the Automatic Matching of Omnidirectional Street Images and Building Data and the Creation of 3D Building Models. *Remote Sens.* **2024**, *16*, 1858. <https://doi.org/10.3390/rs16111858>

Academic Editor: Filiberto Chiabrando

Received: 6 April 2024

Revised: 15 May 2024

Accepted: 16 May 2024

Published: 23 May 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The creation of three-dimensional (3D) building models is an important topic not only in the fields of urban planning, landscaping, disaster management, and urban activity monitoring but also for commercial applications such as movies and virtual reality [1–4]. In recent years, the development of 3D building models has accelerated worldwide with the emergence of digital twins [5,6].

CityGML has been adopted as the data format for several 3D building models [7]. This format is an international standard for 3D models of cities and landscapes and was developed by the Open Geospatial Consortium [8]. A characteristic of CityGML is that it can describe semantic information in addition to geometry. This feature makes the system useful for applications in virtual spaces for advanced urban planning and social science simulations of urban activities.

The level of detail (LOD) is defined in CityGML and divided into five stages, as shown in Figure 1 [9]. LOD0 is simply the building footprint, and LOD1 buildings are represented by simple box models with height information. LOD2 models have additional roof shapes, and LOD3 represents detailed 3D models with façade structures, such as windows and doors. LOD4 takes into account the interior structures of buildings. With this function, it is possible to visualize buildings with appropriate fineness, depending on the scale and zoom

level. Most large-scale 3D building models are limited to LOD12. For example, the Japanese Ministry of Land, Infrastructure, Transport, and Tourism has developed a nationwide 3D building model called PLATEAU. This model implements a textured LOD2 model for some urban center areas, whereas most of the remaining buildings have an LOD of one. However, LOD1–2 building models are insufficient when a detailed building geometry is required for accurate urban analysis. To create 3D building models with a higher LOD than two, model building façade elements such as windows and doors need to be modeled, which used to be completed manually and is extremely labor-intensive. Therefore, an efficient method for creating large-scale 3D building models is urgently needed.

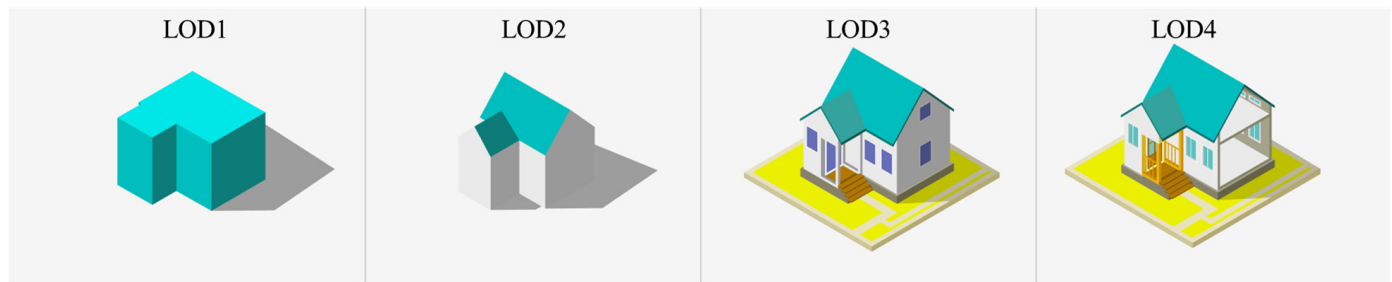


Figure 1. Five levels of detail in CityGML [Reprinted/adapted with permission from Ref. [9]. 2022, MLIT Japan].

One of the biggest challenges in modeling existing buildings is accurately representing the current state of a building, which can include damage and color changes over time. An accurate “as-is” representation of an existing building should take these aspects into account. Many studies have been conducted to create 3D building models by projecting images of buildings extracted from aerial photographs, unmanned aerial vehicles (UAVs), or street-level imagery onto 3D models [10–13]. The model generated by this method may look natural from a distance, but upon closer inspection, the texture resolution may be insufficient, or unnecessary obstacles may be reflected, resulting in poor visualization. This is because the building images are attached directly to the model. Therefore, it makes sense to convert a building image into a natural image that is suitable for generating the texture of a 3D building model.

Recently, several studies have combined data from footprints, light detection and ranging (LiDAR), and street views to create 3D models. This approach combines heterogeneous data to obtain building information from different perspectives to capture buildings in detail. For example, an automated workflow has been developed to reconstruct LOD1 and LOD2 3D building models based on building footprints and LiDAR point clouds [14]. However, these studies have mainly focused on LOD1–2. Studies targeting LOD3 have only experimented with specific buildings as some steps need to be performed manually by users [1,15]. Therefore, a method to automatically generate LOD3 building models for a large area has not yet been developed. To achieve this, it is necessary to use different kinds of data. However, linking multiple data sources is difficult because of the differences in data formats and noise. Complex factors, such as heterogeneity and noise in the data, are cited as challenges for pre-processing. Existing methods are susceptible to image noise and can produce incorrect model shapes if noisy data are used without modification. Therefore, methods that take noisy image data into account are needed.

In addition to the geometric and graphical aspects, object semantics are important for many applications of 3D urban models. For example, the functionality of buildings plays an important role in disaster simulation and mapping. The functionality of a house requires special attention to the semantics of objects, such as the locations of windows and doors. In emergency scenarios, the semantic characterization of objects is required to calculate the extent of flooding that is likely to affect doors and windows or to calculate the risk of failure of doors and windows at a particular location. There are no purely geometric or graphical

models for these applications. Most architectural modeling software, including building information modeling (BIM) software, focuses on new building designs and, therefore, has limited tools for modeling existing building elements.

Given the above, the following problems arise with the current method of creating 3D models: (1) To automatically create detailed models over a large area, high-resolution data from different ground and aerial sources must be linked. However, this task is very complex. (2) Building semantics, such as windows and doors, are crucial for urban planning. However, several existing models do not take these factors into account, resulting in unstructured surface models. (3) It is difficult to “create a 3D model over a wide area” and “reproduce high LOD” at the same time. Models corresponding to LOD3 are created manually one by one.

In this study, these problems have been solved. A framework is proposed for an end-to-end method to create 3D building models with graphical textures and semantics of objects by integrating multi-source data, such as omnidirectional street images, building footprints with attribute information, and aerial photographs. In Japan, various urban data, such as omnidirectional street images with location information and building geographic information system (GIS) footprint data with attribute information, such as building usage and number of floors, have been collected by the government and companies to create digital twins. This study focused on a framework for the entire process, from the pre-processing stage, where multiple raw data sources are combined, to the stage where 3D models of the actual building are created, containing not only relevant geometric and graphical aspects but also the semantics of the objects. Experiments were conducted on urban data from several city blocks in Kobe, Japan, to verify the performance of the proposed system.

The contributions of this study are as follows:

- The main contribution of this study was the automatic linkage of several large raw datasets. The three main inputs in this research were omnidirectional images, building footprints, and aerial photographs. These inputs were automatically linked and pre-processed end to end.
- HRNetV2 was used to extract the wall surface areas and a detection transformer (DETR) was used to extract the locations of the opening of building elements, such as windows and doors. We propose a method to estimate roof types from building images using a CNN and further identify roof colors from aerial photographs. This approach allows the creation of a structured 3D model with accurate semantic information about the building elements.
- With our method, a 3D model with detailed information about window locations, door locations, texture images, roof types, and roof colors can be created automatically and comprehensively. This will contribute to the realization of an automatic LOD for three generations in the future.

2. Related Works

Several studies have been conducted on the reconstruction of 3D building models [16–18]. The data used to analyze and reconstruct urban spaces can be categorized into two main types: camera-captured images and point-cloud data obtained from laser surveys. Because the data can be acquired from both the ground and air, they can be further classified into four categories: ground images, aerial photographs, ground laser point clouds, and aerial laser point clouds. Previous studies have mainly used aerial photographs or aerial laser point clouds [19]. Because these data are obtained from the air, they are useful for estimating information, such as building footprints and roofs that cannot be captured from the ground. In particular, the shape of building roofs can be accurately reproduced using aerial LiDAR [20,21]. In addition, because data can be simultaneously collected over a wide area, there are several use cases for the creation of large-scale LOD1–2 3D building models [22,23]. Large-scale mass models such as the digital surface model (DSM) can be created by using multi-stereo aerial photographs taken from different angles and applying

structure from motion (SfM) and multi-view stereo (MVS) to them [24]. However, data acquired from the air have the disadvantage of their resolution being low. Therefore, they are less suitable for the analysis of building walls than data captured from the ground, and few studies have addressed the modeling of buildings with an LOD of three or more when using airborne LiDAR. The method used in [15] only focused on a single building and not a large-scale city.

In contrast, ground-based images and laser point clouds are well-suited for analyzing building walls because high-resolution information about geographical objects can be easily obtained from these inputs. Various methods have been proposed, such as extracting the structures of windows and doors, considering the symmetry of a building [25], object detection using particle filters [26], and semantic segmentation using recurrent neural networks (RNNs) [27]. Recently, convolutional neural networks (CNNs) have been widely used [28,29], and transformer-based models including the detection transformer (DETR) have emerged [30]. Dore et al. developed a semi-automatic approach for façade analysis of existing buildings using laser and image data; however, they focused only on the façade, not the entire building [31].

Several studies have reconstructed 3D structures of buildings directly from ground images, with SfM and MVS being the most important methods [32,33]. A mesh model created using these approaches tends to be noisy or empty in the regions that are not included in the images. Recently, a new 3D representation using neural networks, known as the neural radiance field, has been proposed to describe cities [34,35]. However, it does not produce solid 3D models, but only street views from arbitrary viewpoints, which makes it difficult to use for simulating urban activities.

In addition, some studies have improved the accuracy of building-side reproduction by fusing models created from ground images with approximate 3D models from aerial photographs [36]. However, these models are created by grouping multiple buildings, making it difficult to distinguish individual buildings. Therefore, a method in [37] was proposed to individually create accurate 3D building models by integrating GIS footprint data and location information of building elements obtained through façade analysis. However, this method uses single colors extracted from image pixels to texture each element of the 3D building model, so detailing the building textures remains a challenge. Texture synthesis methods are useful to solve this problem. Texture synthesis is about creating a natural image that matches the style of the input image. A common method in this field is to create a new image non-parametrically by sampling the pattern information from the original image [38,39]. In recent years, CNN-based methods have become popular, in which an encoder embeds an input image into certain features and a decoder generates images based on these features [40]. They have also been applied to create suitable wall textures from images of rooms [41] and to generate images suitable for pasting onto 3D models [42].

As ground images can be efficiently collected by driving a vehicle equipped with a camera along roads, data collected by various companies and organizations and online viewing services, such as Google Street View, now exist. It is expected that a large amount of image data collected in this way will be used to create large-scale 3D models of buildings. However, to achieve this objective, it is essential to match the buildings in the ground images with the actual buildings. Some studies [43] have linked building images to building footprint GIS data by segmenting the buildings from the images using Mask R-CNN [44] and using directional information. As a semantic segmentation model, HRNetV2 [45] uses a high-resolution network and a low-resolution network in parallel and has shown high accuracy. However, no method has yet been developed that extends the matching method to the creation of 3D building models.

3. Data

3.1. Data and Focus Area

3.1.1. Omnidirectional Images

We used 7968 omnidirectional street-view images from Kobe City, Japan, obtained from Zenrin Corporation. The data were captured with a panoramic camera at 2.5 m intervals, and all data were collected in 2013 (Figure 2 (red point)). The image size was 5400 pixels (width) $\times$ 2700 pixels (height). The information on the recording points and the moving directions (longitude, latitude, vehicle azimuth, and shooting time) was recorded with a GPS device.

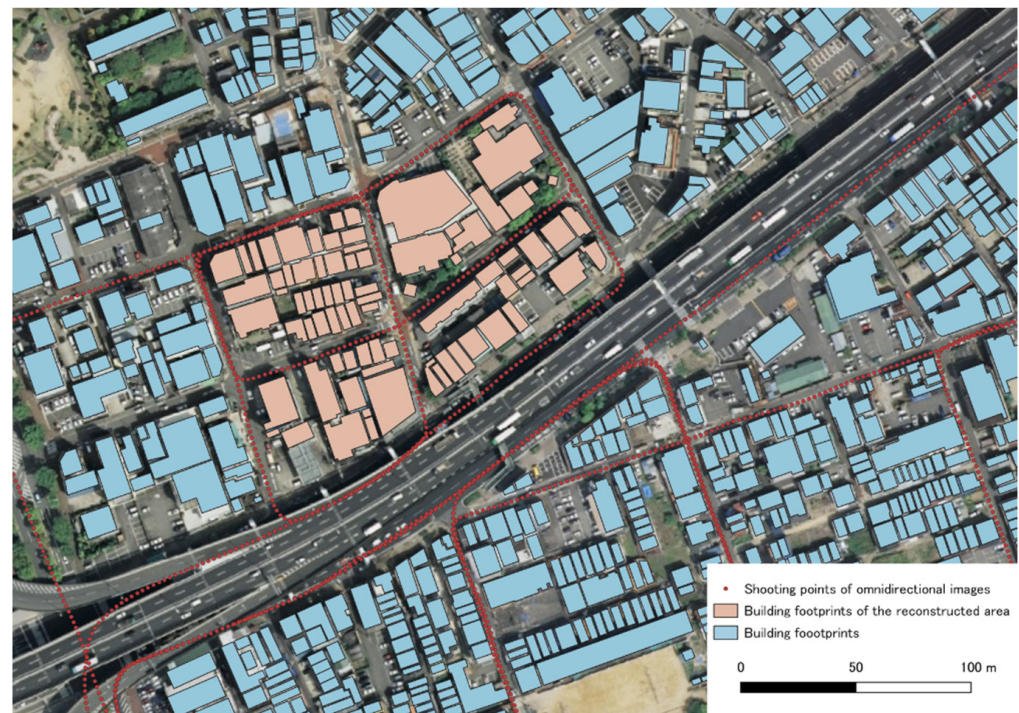


Figure 2. Building data used for this study (shooting points of omnidirectional images, building footprints, and aerial photographs). In our study, approximately, 100 m $\times$ 100 m of the orange-colored buildings were reconstructed.

3.1.2. Building Footprint GIS Data

The Basic Urban Planning Survey of Kobe (2015) provided by the G-Space Information Center (<https://front.geospatial.jp/> (accessed on 1 January 2022)) for GIS data on building footprints in the shapefile format was used (Figure 2). This survey was conducted in cooperation with the prefectural and local governments to understand the use of urban space. Each polygon contains attribute information, such as the building ID, building use, and number of floors.

3.1.3. Aerial Photographs

Seamless aerial photographs provided by the Geospatial Information Authority of Japan were used for this study. A basic electronic national land map (ortho-image) recorded in 2009 was used for Kobe (Figure 2). The ground resolution is approximately 0.40 m.

3.1.4. Texture Images

Texture images were prepared for training the texture synthesis model (Figure 3). In this study, 491 open-source texture images were downloaded from online image libraries (<https://www.freepik.com/>, <https://www.pexels.com/> (accessed on 1 January 2022)). The images correspond to one of the three material classes (brick, wood, and plaster) as the

reconstruction focused on a residential area in Japan, and most of the building surfaces were constructed from materials from these classes.



Figure 3. Examples of texture images.

3.2. Creation of the Dataset

In this study, we divided the above data into (1) data of the target area for the creation of 3D building models, and (2) data for the training and validation of the framework.

3.2.1. Data of the Target Area for the Creation of 3D Building Models

In this study, an area of approximately 100 m square in Kobe City was selected as the target area for constructing the 3D building models. By applying the process described in Section 4.1 to 474 omnidirectional images taken in this region, 78 façade images were obtained from the front side and assigned to each side of the buildings individually. The results are presented in Section 5.1.

3.2.2. Data for Training and Validation of the Framework

A dataset was also created to train and validate the framework of the proposed method. A total of 1233 rectified building images were prepared from the remaining 7524 omnidirectional images. The data were manually annotated to train each part of the framework. For the roof-type classification, 1233 building images were prepared and labeled according to different roof types (flat, gable, or hip). The data for the semantic segmentation model was prepared by labeling some of the building images with 19 different object classes, and 356 images were created. To recognize the positions of windows and doors, the data were divided into three classes (windows, doors, and walls), and 909 building images were created. To train the texture synthesis model, the texture images described in Section 3.1.4 were resized to 512×512 pixels and assigned a material class (wood, plaster, or brick). In this study, 75% of the data were used for training, 10% for validation, and 15% for testing.

4. Method

In this study, a framework for the automatic creation of 3D building models is proposed, as shown in Figure 4. The input data include omnidirectional street-view images, building footprints with attribute information, and aerial photographs. The framework is divided into five steps, which are explained later in this section.

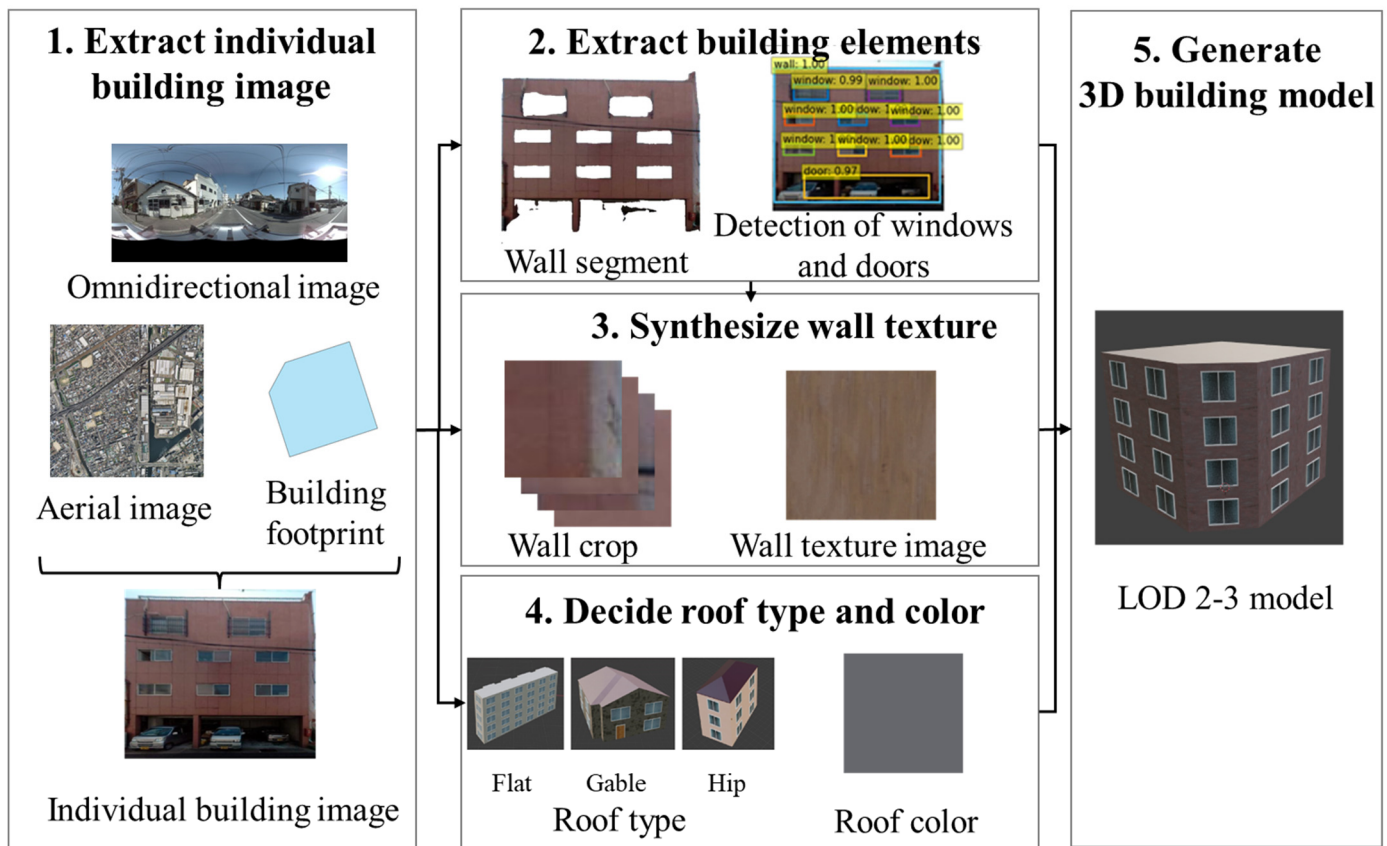


Figure 4. Overview of the proposed framework.

4.1. Matching Omnidirectional Images and Buildings

4.1.1. Matching Building Bounding Boxes and Building Footprints

Omnidirectional images and building footprints can be displayed on a GIS platform, as shown in Figure 5. The polygons of the buildings near the shooting points were extracted, and the ranges of azimuth angles from the shooting points (direction with true north set to 0°) for each building polygon were calculated. The angle of each building with respect to the shooting point was calculated as follows:

$$\theta_i^{l,r} = \tan^{-1}(x_i^{l,r} - x_c, y_i^{l,r} - y_c) \quad (1)$$

where (x_c, y_c) are the latitude and longitude of the shooting point, and the latitude and longitude of the vertex (l, r) of building i obtained from the GIS building footprint data are $(x_i^{l,r}, y_i^{l,r})$. In addition, the range of existence of the building (θ_i^l, θ_i^r) was determined, based on the direction of travel, by subtracting the azimuth angle of the direction of travel of the vehicle from the azimuth angle range of each building. Of the buildings included in the omnidirectional image, those that were directly adjacent to the shooting point were captured from the front. However, other buildings were captured along an oblique angle, making it impossible to obtain accurate information about the building's façade. Therefore, for each omnidirectional image, it was desirable to select and use only the buildings closest to the direction perpendicular to the direction of travel. Therefore, the center $\theta_i^m = (\theta_i^l + \theta_i^r)/2$ was calculated for each range of existence of the building, and the building with θ_i^m closest to 90° or 270° was selected as the building that was at the center of a particular omnidirectional image.

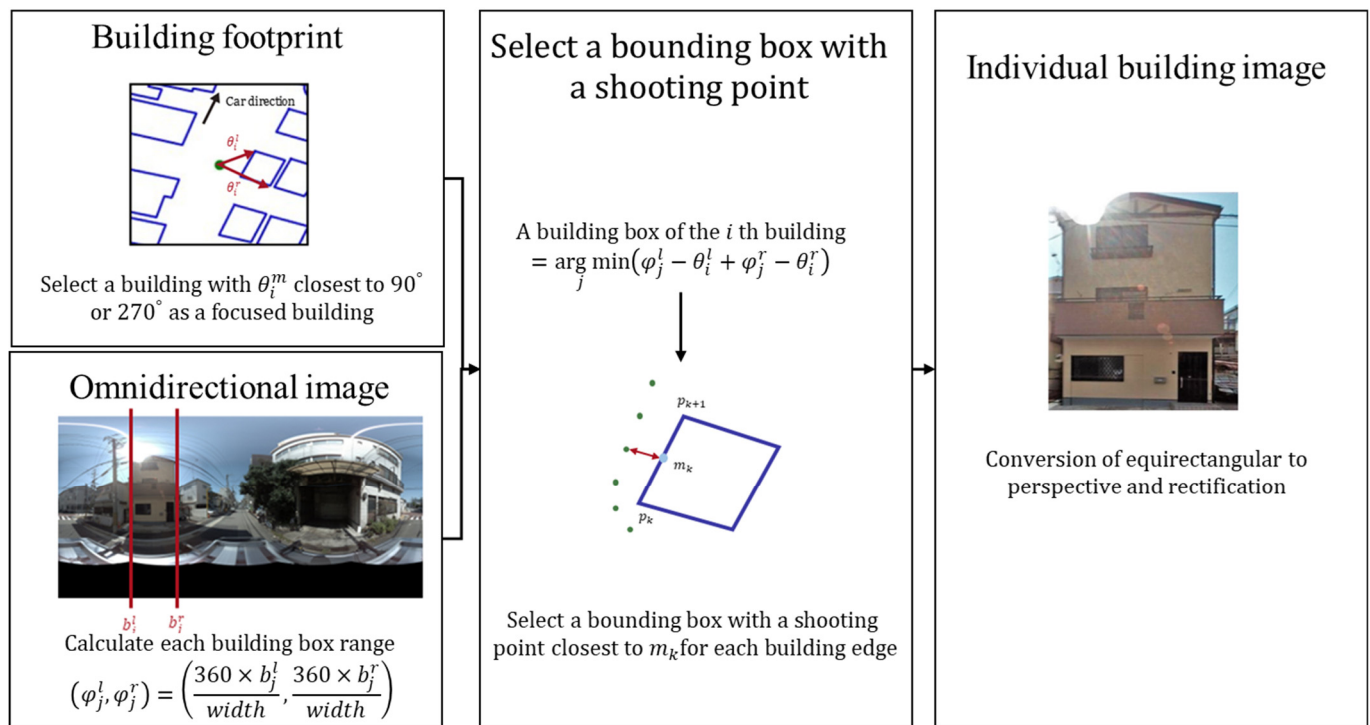


Figure 5. The flow of matching omnidirectional images and buildings.

Next, the bounding boxes of the building were extracted from the omnidirectional images. In this case, Mask R-CNN [44] was used, which was previously trained with the common objects in the context (COCO) dataset [46] and fine-tuned using omnidirectional images. The number of pixels was denoted and counted from the left side of the omnidirectional image to the left and right edges of the j -th bounding box as b_j^l and b_j^r . The omnidirectional image had a 360° view of the scene from the shooting point, and the left and right sides of the image corresponded to the travel direction. Thus, the building range of the j -th bounding box relative to the direction of travel $(\varphi_j^l, \varphi_j^r)$ is represented as follows:

$$(\varphi_j^l, \varphi_j^r) = \left(\frac{360 \times b_j^l}{width}, \frac{360 \times b_j^r}{width} \right), \quad (2)$$

where *width* is the number of horizontal pixels in the image.

This building range $(\varphi_j^l, \varphi_j^r)$ was compared with the GIS-based range of existence (θ_i^l, θ_i^r) , and the pair with the closest value was searched for as follows:

$$\text{bounding box of } i\text{-th building} = \arg \min_j \left\{ (\varphi_j^l - \theta_i^l + \varphi_j^r - \theta_i^r) \right\}. \quad (3)$$

Consequently, one or more building bounding boxes were linked to each building's footprint polygon.

4.1.2. Matching Building Bounding Boxes and Building Edges

Next, the side of the actual building to which each bounding box corresponded was determined. It was assumed that the building wall captured in the building bounding box was closest to the shooting point of the omnidirectional image. As described in Section 4.1.1, several bounding boxes were tied to the building polygon. Given that the k -th vertex of a building footprint polygon is p_k , the midpoint of the k -th edge can be computed as $m_k = (p_k + p_{k+1})/2$. The distance between the shooting points of the omnidirectional

images containing each bounding box and the midpoint m_k of each edge was calculated and matched the edge that minimized the distance to the bounding box.

Using this method, multiple bounding boxes can be linked to a single edge. To obtain a one-to-one correspondence between the edges and the bounding boxes, bounding boxes that were not closest were excluded. Some building walls were occluded and were not captured in the omnidirectional images. The edges of such walls were defined as empty conditions as the bounding boxes could not be linked to them.

4.1.3. Rectification

Omnidirectional images are equirectangular. Therefore, building images cut directly from them are highly distorted, especially at the top and bottom. Therefore, a transformation into a perspective image was performed, so that the image was suitable for a façade analysis. In addition, because the buildings in the transformed image were tilted or obliquely captured, the image was also transformed into a homograph to bring it into a frontal orientation.

Although the matching and rectification described above were performed automatically, the process did not work well for complex buildings where the wall boundaries were difficult to identify. In such cases, we manually corrected image ties and rectifications.

4.2. Building Element Extraction

One building image for each building edge was used to create wall crops and recognize the positions of windows and doors.

4.2.1. Wall Crop Creation

A semantic segmentation of the building images was performed to extract the wall surface areas. To create crops of wall surfaces that were free of additional objects, 19 different types of class labels were defined (e.g., walls, windows, doors, or roofs) that appeared in the building image at a certain frequency. In the experiments with street images, HRNetV2 [45] was used, which has already shown high accuracy, as a semantic segmentation model. In our study, a model pre-trained on the large image dataset ADE20K [47,48] was used and fine-tuned using images of buildings in Kobe City. The cross-entropy loss between the predicted and correct labels was used for training.

The areas predicted as walls by semantic segmentation were extracted, and ten images (128×128 pixels) were randomly cropped from the wall surface segments. Multiple crops were prepared because a single crop may accidentally select a region that includes objects that do not belong to a wall.

4.2.2. Window and Door Location Detection

To identify the locations of windows and doors in a 3D building model, object detection for three classes of objects (windows, doors, and walls) was performed, and the locations of windows and doors relative to wall surfaces were calculated. A detection transformer (DETR) [30] was used as the object detection model, which also fine-tuned the pre-trained model with the building images. The losses of the bounding boxes and the labels were used as training losses. The loss of the bounding boxes was calculated by combining the generalized intersection over union (IoU) loss [49] and the L1 loss, whereas cross-entropy loss was used for the loss of the labels.

4.3. Texture Image Generation

A wall crop contains a large amount of noise. Therefore, a new image from the wall crops was created, which was suitable as a texture model for the 3D building model. As shown in Figure 6, a texture synthesis model with an encoder–decoder structure was constructed by referring to an existing method [41]. The base color was calculated by averaging each pixel value in the crop I_i and subtracting the base color c_i to separate the pattern image p_i^{org} . This was input to the encoder E and embedded into the feature vector

$\vec{t} = E(p_i^{org})$. This embedding was then passed through the decoder D to generate a new pattern image, $p_i^{new} = D(\vec{t})$. Finally, a new texture image T_i was synthesized by combining this pattern p_i^{new} with the base color c_i . In addition, a material classifier branch was set up to estimate the material class of the target image by passing the embedding through the classifier f . Three classes (wood, plaster, and brick) were prepared for the building wall material. ResNet-18 was used as the main encoder, and a 5-layer CNN was used as the decoder, as reported in a recent study [41].

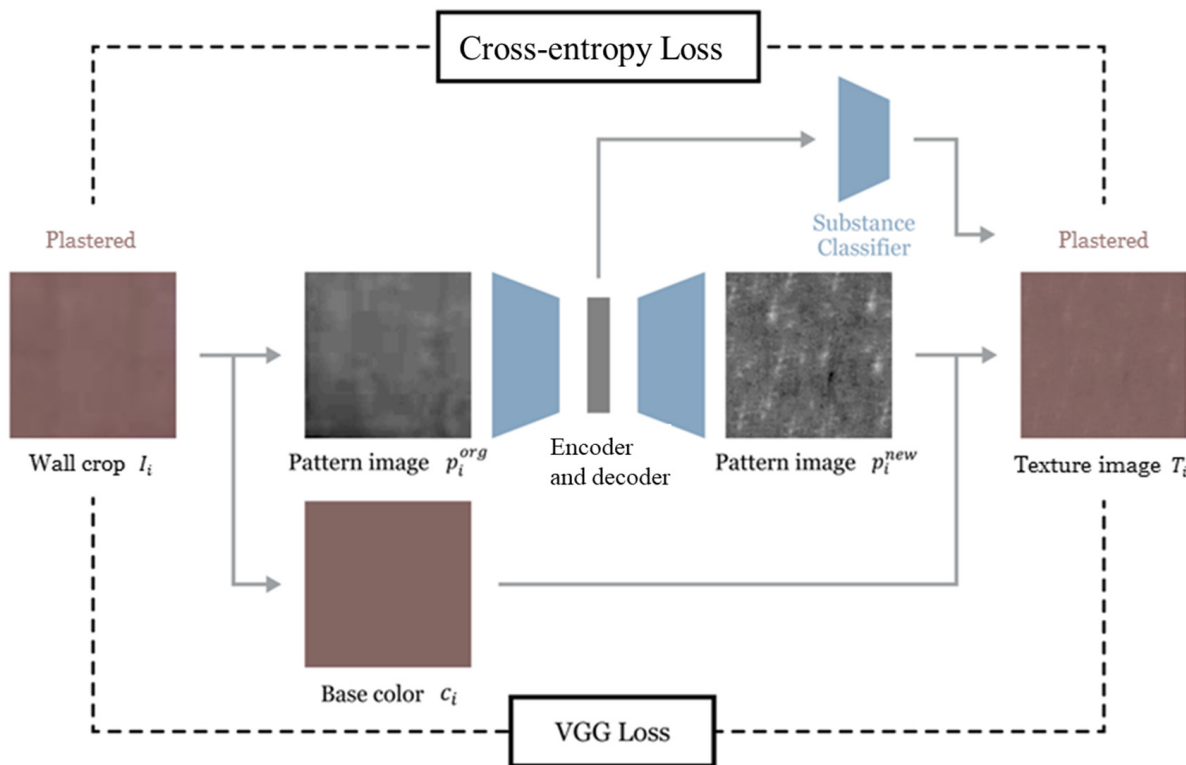


Figure 6. Schematic representation of the texture synthesis model.

The model was trained with the visual geometry group (VGG) loss for the image pixels and the cross-entropy loss for the texture material. The VGG loss was calculated using the mean squared error of the intermediate feature map produced when the cropped and generated images were input into VGG-19 [40,50,51].

As described in Section 4.2.1, 10 wall crops per building image were created and used to synthesize the 10 texture images. The VGG loss between each crop was calculated, and the texture image was synthesized based on this loss. The pair whose loss value was closest to the average of the 10 loss values was selected as representative.

The texture images generated using this method were not seamless. Embark Studios [52,53] was used as a post-processing step to create tunable texture images. In this study, texture images were only synthesized for wall surfaces and not for windows or doors. The reason for this was that windows and doors made of glass often contain reflections, which make it difficult to create suitable crops.

4.4. Determination of Roof Type and Color

4.4.1. Roof Type Classification

The roof type was determined based on the image classification of each building image. Three roof types were used in this study: flat, gabled, and hipped (Figure 7). Buildings have many different roof shapes. However, because our method was based on a single ground image, the problem was simplified by focusing on the three basic roof shapes mentioned

above. The CNN model used was VGG16 [51], which was pre-trained in ImageNet [54] and fine-tuned using building images of Kobe City. The cross-entropy loss between the predictions and the correct labels was used for training.

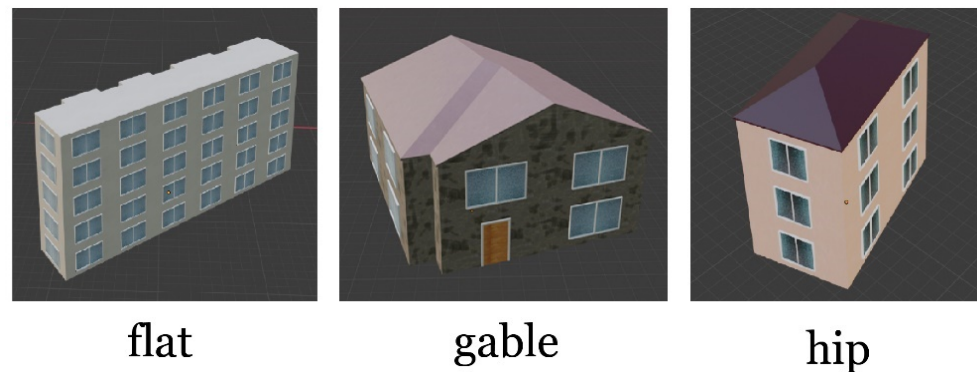


Figure 7. Types of building roofs.

4.4.2. Roof Color Determination

The roof color was determined by extracting the pixels of each building from the aerial photographs. We overlaid the aerial photograph and the GIS footprint of the building and cropped an image of each polygon. However, owing to location errors, there were slight discrepancies between the actual roof area and the footprint. Therefore, the segment contained not only the roof but also roads or vegetation. Therefore, simply averaging the pixel values of the segment was not sufficient to determine the exact color of the roof, and the k-means++ method [55] was used to cluster the pixels. The number of clusters k was set manually. Each segment was checked manually, and it was confirmed that in most cases the segment consisted of three elements: the main color of the roof, the color of the shadows and objects on the roof, and the color outside the building area, such as roads. Therefore, clustering was performed with $k = 3$ and the color of the largest cluster was determined as the roof color.

4.5. Three-Dimensional Building Model Generation

We used Blender 2.92 to model the 3D buildings. Almost all operations in the GUI of Blender can be written in Python, which allowed us to automate the entire modeling process.

First, a base 3D model was created by extruding a footprint polygon based on the height of the building. The building height was calculated by multiplying the height of each story by the number of floors. The attribute information linked to the building footprint included the number of floors and the building use. The height of a story was determined based on the building usage (for example, “residential” was 3 m, and “business and commercial” was 4 m). The roof shape of the model was transformed based on the roof type determined in Section 4.4.1, using Blender’s building tools.

The locations and sizes of the building elements were identified as explained in Section 4.2. The texture image created in Section 4.3 was mapped for the wall texture of each object. Special texture images prepared in advance were used for the windows and doors. For the roof texture, the roof color extracted from the aerial photography was used.

Some walls were indicated as blanks because they were occluded in the omnidirectional image, and other building images could not be captured. For such walls, the locations of the building elements are unknown, and the above process cannot be used. Therefore, we propose a modeling method that uses the surface of the surrounding wall as a reference. The wall texture of the blank side was replaced with a random selection of texture images of one of the other walls in the same building. For the window size, a simple average of all window sizes in the building was used. The window locations were set such that the space between the windows was half the window size, and the windows were arranged on each floor.

5. Results and Discussion

Inference processes, such as semantic segmentation, were performed using an AWS EC2 g4dn instance. For the building modeling, the model was generated in four separate components, as the simultaneous reproduction of all 81 buildings included in the experimental area would have led to an unreasonably high load. Each part of the generation process required approximately 2 h, and a complete model was created in approximately 8 h.

5.1. Matching Building Images and Building Edges

Following the flow shown in Figure 5, the building images were matched with the building edges. Figure 8 shows an example of a building with ID 314960. The images were correctly assigned to each side, whereas the sides occluded by the road were shown as blanks. This study focused on 81 buildings in Kobe. Nineteen of them were fully occluded and all sides were blank, which meant that the available images covered 76.5% of the buildings selected for this study. All sides, except the blank ones, were visually inspected to ensure that the associated images matched correctly. The accuracy of the matching was 86.9%, demonstrating the effectiveness of this method.



Figure 8. Examples of images linked to edges of a building polygon with ID 314960.

Upon closer examination of the results, the matching accuracy for building edges along straight roads was 91.2%, whereas the accuracy for buildings at turning corners was 79.3%. This was owing to the change in the direction of the vehicle camera at the corner. The direction of travel of the car significantly changed at a turn, and the target building was usually offset 90° to the left or right, making a mismatch more likely. As described in Section 4.1.3, the mismatch between building edges and building images prevented the correct placement of windows and doors; therefore, manual modifications were made.

5.2. Building Element Extraction

5.2.1. Wall Crop Creation

The semantic segmentation of the building image resulted in a pixel-based accuracy as high as 88.3%. Figure 9 shows the visual results: on the left is the original image, in the middle is the annotation data, and on the right is the prediction. In the predicted images, the wall surfaces were segmented in gray, and all were accurate. In the classes without the wall (for example, in the first row), the windows were sparsely distributed. This was presumably because the reflections on the glass were similar and could not be identified as windows.

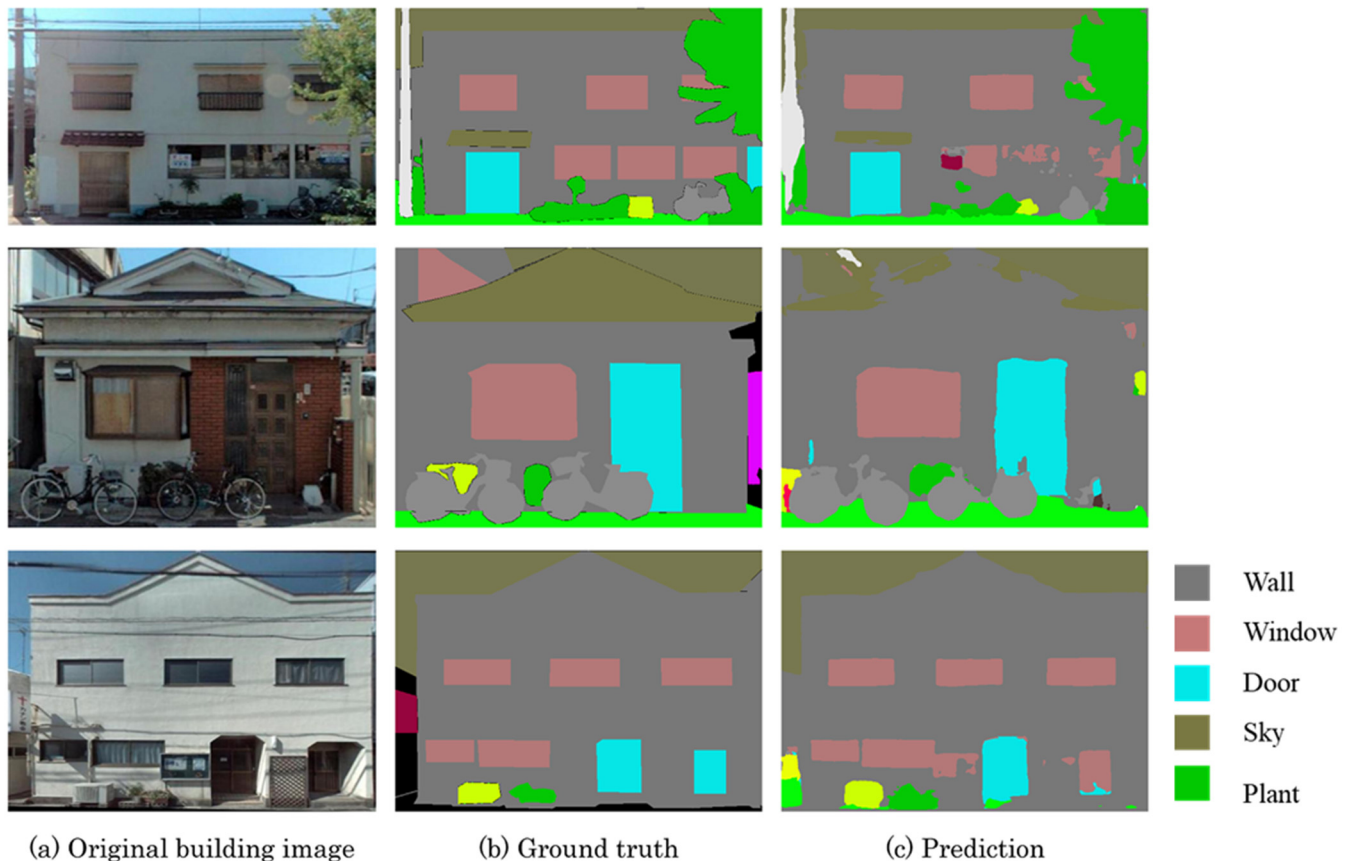


Figure 9. Results of the semantic segmentation: (a) original building image, (b) ground truth, and (c) prediction.

An evaluation using IoU segmentation was performed, which indicated the extent to which the predicted mask overlapped with the correct mask. This metric is considered suitable for evaluation in cases where contours need to be extracted precisely. The wall and sky classes had high IoU values of 0.875 and 0.943, respectively. Both had a particularly high frequency of occurrence and a high percentage of pixels in the image; therefore, the model fit well with the detection of these classes. However, the scores for windows and doors were relatively low at 0.608 and 0.527, respectively, even though these are the most important elements included in almost all images, such as the walls and sky. This may be due to a certain number of false recognitions and reflections on the glass, which could have affected the IoU.

5.2.2. Window and Door Location Detection

The results of the window and door detection were evaluated using the average precision (AP) metric, which is the value obtained by integrating the precision–recall curves and averaging them across class labels. This metric was chosen to make a comparison with

the DETR. The result was an AP of 0.422, whereas the study that originally proposed the DETR framework had an AP of 0.420. Our result was judged to be acceptable since it was almost equal to or slightly better than the DETR value.

Figure 10 shows an example of this detection method. It is noticeable that areas obscured by laundry or blinds were correctly recognized. This may be because the model can infer the entire image from a partially visible window as the training dataset was annotated in this way. In general, the windows and doors of buildings in Europe are often uniformly arranged and have similar shapes, whereas the shapes of the façades greatly varied in the Japanese buildings considered in this study. The usefulness of this method was, therefore, demonstrated by its ability to recognize image data with a certain degree of accuracy.



Figure 10. Results of window and door location detection: (a) original building image and (b) prediction.

5.3. Texture Image Generation

Figure 11 shows the results of the texture synthesis. In both cases, images were generated with colors comparable to those of the cropped images, confirming that the base color was preserved. In the first row, the crop contains the shadows of the electrical wires, which disappear after synthesis, indicating that only a small amount of noise can be ignored. In comparison with the actual building image on the left, the textured image was generated as a wall surface without any discrepancies. However, in the case of the second row, although the images before and after synthesis are similar, upon closer examination, it

becomes evident that the vertical patterns on the tin walls have been converted to a wood grain pattern. This may be because the training dataset contained numerous brown wood texture images. In the third row, the synthesized image is almost monochrome, whereas the original building image shows that the walls are tiled. This may be because the original architecture was recognized as noise and smoothed out by the model. In addition, this tiled pattern is not visible in the wall crop. Therefore, the inadequate resolution of the image may also be a factor in the conversion error.

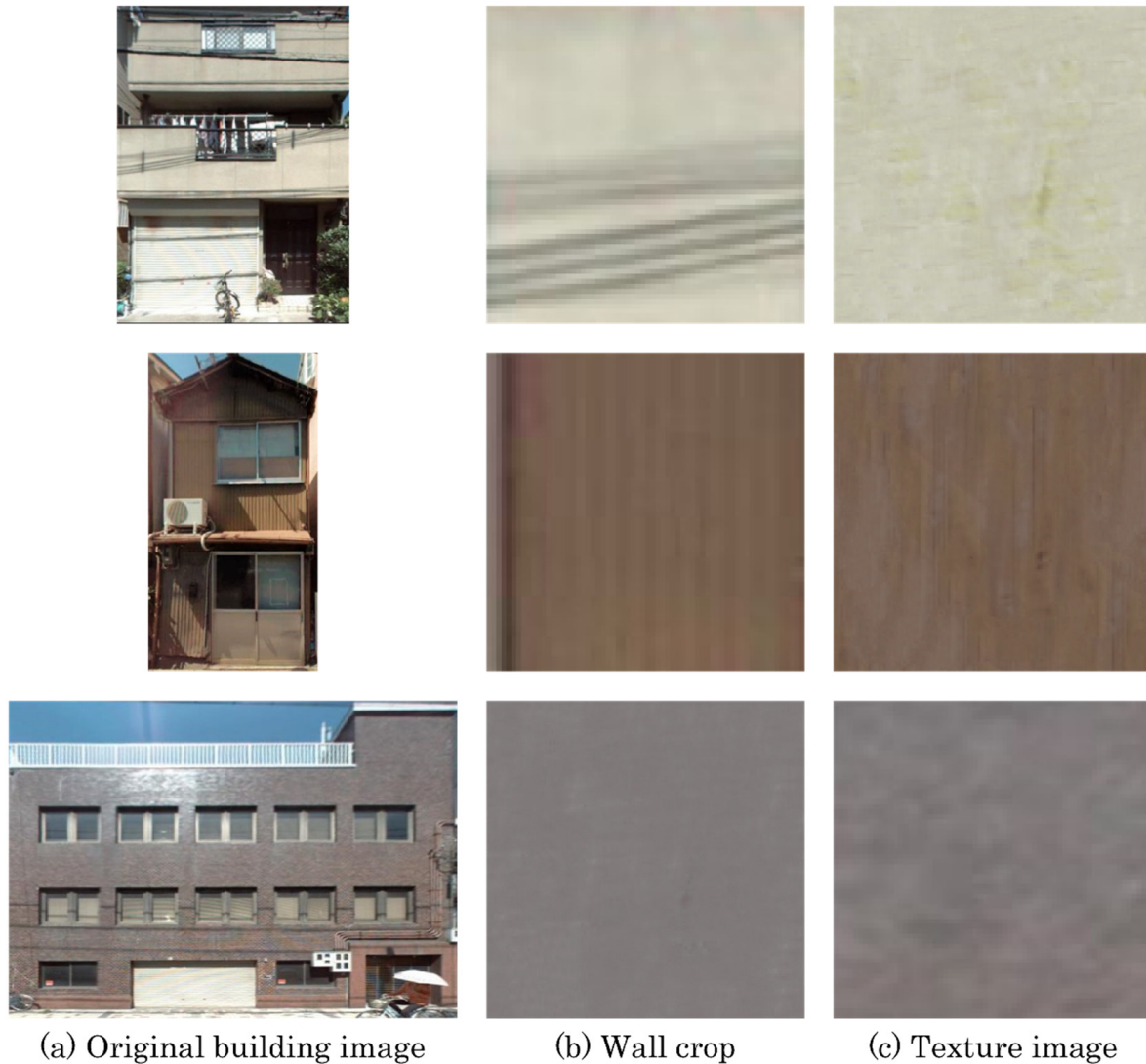


Figure 11. Results of the texture synthesis: (a) original building image, (b) wall crop image, and (c) synthesized texture image.

5.4. Determination of the Roof Type and Color

5.4.1. Roof Type Classification

The classification of building images by roof type resulted in a high accuracy of 89.7% for the test data (Table 1). The accuracy by class was 95.6% for flat roof type, 88.5% for gable, and 87.4% for hip. It was confirmed that there was some confusion between the gable and hip roofs. The failure cases were checked, and it was found that almost all cases were difficult to determine even by the human eye. Therefore, better accuracy would be achieved if the outlier data were removed during the pre-processing of the dataset.

Table 1. The result of roof type classification.

Roof Type	Accuracy
Flat	95.6%
Gable	88.5%
Hip	87.4%
Overall	89.7%

5.4.2. Roof Color Determination

Figure 12 shows the results of the extraction process. For example, in the first row, there is a green area in the polygon because a tree covers the roof. Therefore, if all pixels were averaged, the result would be a greenish color; however, the k-means++ method was able to extract the main reddish-brown color. The second row also contains the shaded areas in the polygons, which resulted in a dark gray color when averaged. However, the k-means++ method could extract the bright gray color.

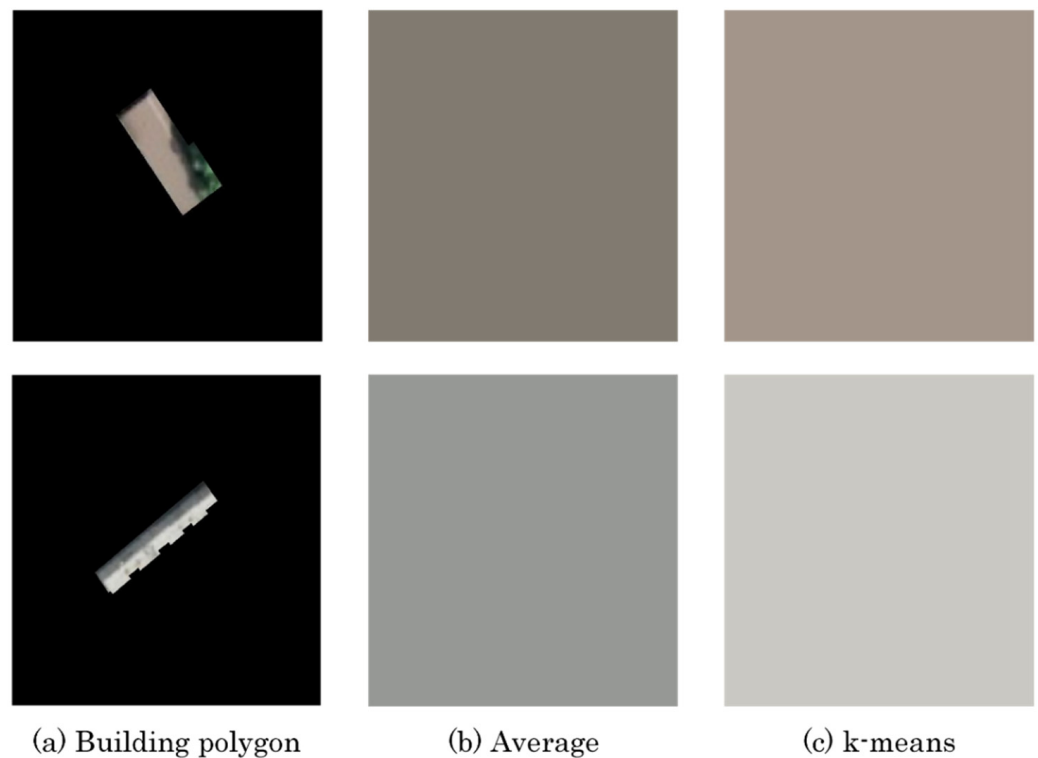


Figure 12. Results of roof color extraction: (a) building polygon cut from an aerial photograph, (b) average color of the polygon, and (c) extracted color using the proposed method.

5.5. Three-dimensional Building Model Generation

We visually inspected the geometry of the segments, and 86.2% of the reconstructed buildings correctly represented the geometry of the windows and doors. The remaining had problems with the location and geometry of the doors and windows. The main causes were glass doors that were not recognized as doors and created as windows, and objects that were partially obscured by trees or fences that were not properly reflected in the 3D model.

Figure 13 shows the generated 3D and Google Earth models of the area. A comparison confirmed that the roof shape obtained using the proposed method was significantly different from the shape determined in Google Earth. For example, although it was correctly determined that a building had a gable roof, the direction of the ridge was not recognizable, resulting in different roof shapes. Compared with the latest studies, methods

using aerial images or LiDAR can reproduce a variety of roof shapes of buildings over a wide area and are more accurate than our method [21,24]. The disadvantage of our study is that it cannot represent complex roof shapes because it classifies them into three types.

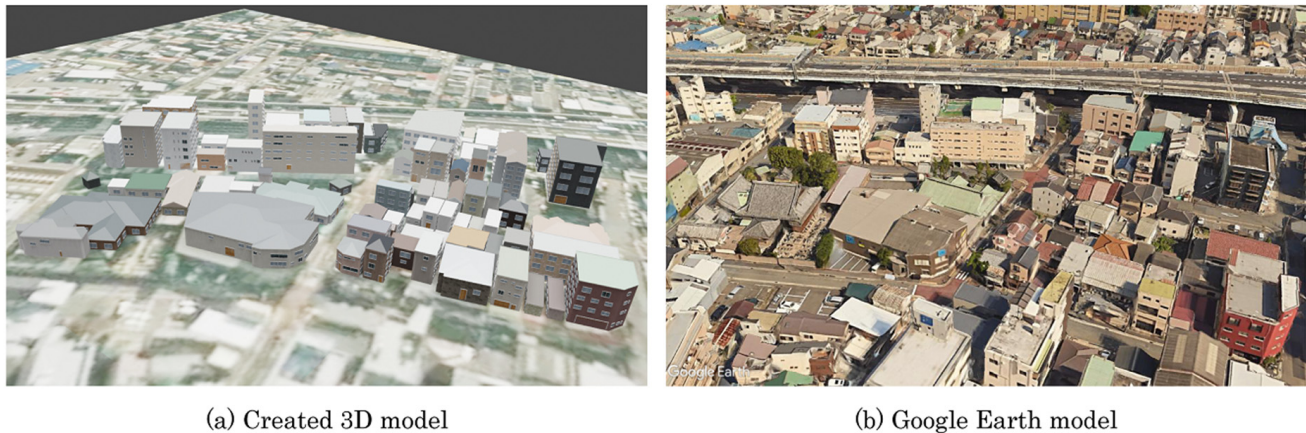


Figure 13. Comparison of the created 3D model with the Google Earth model of the target area.

The individual cases are shown in Figure 14. In the first case, almost all windows on both sides of the building were correctly located. Because the 3D model was extended from the box-shaped LOD1 model, there was no noise on the building surface compared with the method of 3D reconstruction from the ground image using SfM, and the semantics of the building elements were recognizable. A conventional mass model cannot accurately reproduce the details of the walls, whereas our model can be expected to more accurately reproduce the volume of a building as it takes into account the building elements, leading to more accurate disaster simulations. In addition, the wall textures accurately reflected the actual red wall pattern, indicating that the wall cropping and texture synthesis were accurate. Compared with recent image-based building reconstruction methods, our model is superior in that it reconstructs the geometry of windows and doors with comparable accuracy and also generates natural textures of buildings [37,56]. In these studies, the building texture was either not considered or only applied a single color extracted from a building image to the wall of the 3D building model; therefore, it can be said that our 3D models have more realistic building textures.

However, as mentioned earlier, there is an uncertainty in our model regarding the roof shape. For example, the roof of a building in Google Earth had a two-story shape, whereas our system only provided a simple flat roof shape. Therefore, it was not possible to correctly render such complicated shapes. This problem could be partially solved by increasing the number of roof-type classes. In addition, compared with studies that used both ground images and LiDAR, our model is inferior in terms of the accuracy of the location and shape of individual building objects [31]. However, building parts such as roofs were not sufficiently validated in this study. Therefore, it is expected that more accurate and comprehensive building models can be constructed if inputs such as ground laser point clouds are added to the proposed method.

In the second case, the walls of the building have different colors on different sides. In our method, the building images were linked to each side of the building so that the generated model could represent such differences in the wall texture. The front right side of the building has the shape of a balcony, but our system does not support such a complex shape. Therefore, the model resulted in a monotonous structure. To achieve LOD3, it will be necessary to be able to recreate such special geometries for building exteriors in the future.

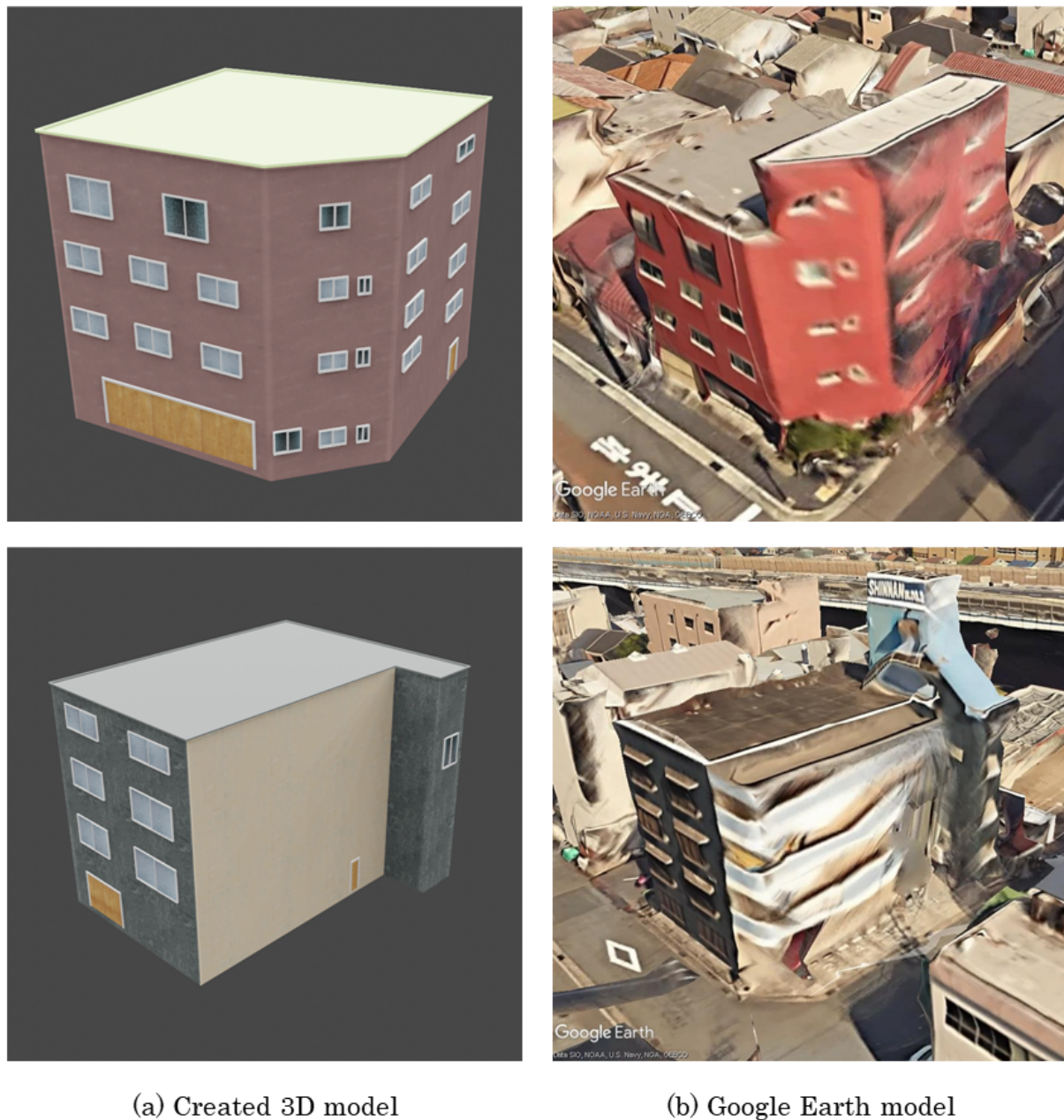


Figure 14. Individual examples of generated 3D models: (a) created 3D model and (b) Google Earth model.

5.6. Confidence Score for 3D Building Models

If the confidence score is set to indicate the accuracy of a 3D building model, we can numerically determine whether the analysis results are reliable for various applications, such as urban analysis and simulations. For example, a 3D building model created with SfM can reconstruct multiple buildings simultaneously; therefore, it is impossible to determine the accuracy of individual buildings. Because numerical values, such as the accuracy of recognizing window positions for each building, are calculated in our framework, it is possible to link such confidence scores to individual 3D models. In cases where no ground-truth data are available, it is difficult to make a judgment about the accuracy.

However, it is possible to evaluate the confidence in the inference based on the variance in the probability of each class derived from the SoftMax function at the time of inference. Thus, our method can be considered useful for the functionality of 3D building models.

5.7. Limitation

Our method has several limitations. As mentioned previously, our method uses only three roof types, which leads to uncertainties in the roof shape of the building model. In addition, although not included in the experimental area of this study, non-planar buildings such as towers could be reproduced up to LOD2. However, it was difficult to identify the positions of windows and doors, similar to the high-rise floors of skyscrapers because of their typical image resolutions. As the windows and doors in this study were positioned based on a single image, the 3D structure (protrusion) of each object could not be reproduced. More accurate modeling could potentially be achieved by introducing new inputs, such as LiDAR, multi-stereo aerial photographs, or approximate mass 3D models.

Moreover, although multi-source urban data were used as the input, the timing of data collection may have led to inconsistencies. For example, cases can arise in which some buildings present in aerial photographs are not shown in street photographs. To address this problem, an attempt should be made to coordinate the timing of data collection.

We used omnidirectional images, building footprint GIS data, and aerial photographs as inputs in this study. However, designing an LOD4 model that represents the interior structures of buildings using these inputs is challenging. We expect that LOD4 can be realized by matching the data of the external structure of a building created by this research with the data of the internal structure of a building prepared separately.

6. Conclusions

In this study, a framework was developed to automatically generate 3D building models by combining multiple data sources such as omnidirectional images of streets, building footprints with attribute information, and aerial photographs. The extracted building image was matched with each building edge in the GIS by referring to the location information and the shooting direction of the GIS. Semantic segmentation and object detection were performed on the building image to extract wall segments and identify the locations of windows and doors. New wall texture images were synthesized from the wall segments, and the type and color of the roof were determined via the image classification of building images and pixel extraction from aerial photographs. Finally, textured 3D building models are generated by combining the building information acquired during each part of the process. An experiment was conducted using urban data from Kobe City, and numerically evaluated the intermediate results for each component. A high accuracy was obtained for each process. Finally, we visually confirmed that the 3D models developed using the proposed method reflect the detailed façade information of actual buildings with a certain degree of accuracy, demonstrating the validity of the proposed method.

The contributions of this study include the matching of multiple data sources and a method for reproducing the semantics of detailed building elements in a 3D model. By applying this method, we anticipate that a method to automatically generate LOD3 buildings from multiple raw open data sources will be developed in the future. As the research data are based on street and satellite images that are readily available across all countries, the framework proposed in this study can be used overseas. One of the future works will be to increase the variety of roof shapes that can be reproduced by increasing the number of classes to classify the roof types or by introducing new inputs, such as LiDAR, multi-stereo aerial photographs, or approximate mass 3D models. The development of a roof shape generation model that utilizes existing LOD2 3D building models as training data may lead to more precise roof shape representation. Another objective for the future is to enable the fast and accurate creation of 3D models for larger areas. To achieve this, the performance of certain processes needs to be further improved by increasing the number of training images, fine-tuning the hyperparameters, and simplifying the network model.

Author Contributions: Conceptualization, Y.O. and Y.S.; methodology, R.N., G.S. and H.M.; software, R.N. and G.S.; validation, R.N. and G.S.; formal analysis, R.N., G.S. and H.M.; investigation, R.N., G.S. and H.M.; resources, Y.O. and Y.S.; data curation, R.N. and G.S.; writing—original draft prepa-

ration, R.N. and G.S.; writing—review and editing, G.S. and Y.O.; visualization, R.N.; supervision, Y.S.; project administration, Y.S.; funding acquisition, Y.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by JSPS KAKENHI Grant Number 22K04490.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

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

References

1. Gruen, A.; Schubiger, S.; Qin, R.; Schrotter, G.; Xiong, B.; Li, J.; Ling, X.; Xiao, C.; Yao, S.; Nuesch, F. Semantically enriched high resolution LoD 3 building model generation. *ISPRS Photogramm. Remote Sens. Spat.* **2019**, *42*, 11–18. [CrossRef]
2. Kolbe, T.H.; Gröger, G.; Plümer, L. CityGML: Interoperable access to 3D city models. In *Geo-Information for Disaster Management*; Oosterom, P., Zlatanov, S., Fendel, E.M., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 883–899. [CrossRef]
3. Wang, R. 3D building modeling using images and LiDAR: A review. *Int. J. Image Data Fusion* **2013**, *4*, 273–292. [CrossRef]
4. Zlatanov, S.; Van Oosterom, P.; Verbree, E. 3D Technology for Improving Disaster Management: Geo-DBMS and Positioning. In Proceedings of the XXth ISPRS Congress; Available online: <https://www.isprs.org/PROCEEDINGS/XXXV/congress/comm7/papers/124.pdf> (accessed on 1 January 2024).
5. Arroyo Otori, K.; Biljecki, F.; Kumar, K.; Ledoux, H.; Stoter, J. Modeling cities and landscapes in 3D with CityGML. In *Building Information Modeling*; Springer: Cham, Switzerland, 2018; pp. 199–215. [CrossRef]
6. Oishi, S.; Koide, K.; Yokozuka, M.; Banno, A. 4D Attention: Comprehensive framework for spatio-temporal gaze mapping. *IEEE Robot. Autom.* **2021**, *6*, 7240–7247. [CrossRef]
7. Soon, K.H.; Khoo, V.H.S. CityGML modelling for Singapore 3D national mapping. *ISPRS Photogramm. Remote Sens. Spat.* **2017**, *42*, 37–42. [CrossRef]
8. Gröger, G.; Plümer, L. CityGML—Interoperable semantic 3D city models. *ISPRS J. Photogramm. Remote Sens.* **2012**, *71*, 12–33. [CrossRef]
9. Ministry of Land, Infrastructure, Transport and Tourism, PLATEAU. Available online: <https://www.mlit.go.jp/plateau/> (accessed on 1 January 2024).
10. Brenner, C.; Haala, N.; Fritsch, D. Towards Fully Automated 3D City Model Generation. In *Automatic Extraction of Man-Made Objects from Aerial and Space Images (III)*; 2001; pp. 47–57. Available online: https://ifpwww.ifp.uni-stuttgart.de/publications/2001/Haala01_ascona.pdf (accessed on 1 January 2024).
11. Lee, J.; Yang, B. Developing an optimized texture mapping for photorealistic 3D buildings. *Trans. GIS* **2019**, *23*, 1–21. [CrossRef]
12. Rau, J.-Y.; Teo, T.-A.; Chen, L.-C.; Tsai, F.; Hsiao, K.-H.; Hsu, W.-C. Integration of GPS, GIS and photogrammetry for texture mapping in photo-realistic city modeling. In Proceedings of the Pacific-Rim Symposium on Image and Video Technology, Hsinchu, Taiwan, 10–13 December 2006; pp. 1283–1292. [CrossRef]
13. He, H.; Yu, J.; Cheng, P.; Wang, Y.; Zhu, Y.; Lin, T.; Dai, G. Automatic, multiview, coplanar extraction for CityGML building model texture mapping. *Remote Sens.* **2021**, *14*, 50. [CrossRef]
14. Tack, F.; Buyuksalih, G.; Goossens, R. 3D building reconstruction based on given ground plan information and surface models extracted from spaceborne imagery. *ISPRS J. Photogramm. Remote Sens.* **2012**, *67*, 52–64. [CrossRef]
15. Wen, X.; Xie, H.; Liu, H.; Yan, L. Accurate reconstruction of the LOD3 building model by integrating multi-source point clouds and oblique remote sensing imagery. *ISPRS Int. J. Geo-Inf.* **2019**, *8*, 135. [CrossRef]
16. Bshouty, E.; Shafir, A.; Dalyot, S. Towards the generation of 3D OpenStreetMap building models from single contributed photographs. *Comput. Environ. Urban Syst.* **2020**, *79*, 101421. [CrossRef]
17. Diakite, A.A.; Zlatanov, S. Automatic geo-referencing of BIM in GIS environments using building footprints. *Comput. Environ. Urban Syst.* **2020**, *80*, 101453. [CrossRef]
18. Otori, K.A.; Ledoux, H.; Biljecki, F.; Stoter, J. Modeling a 3D city model and its levels of detail as a true 4D model. *ISPRS. Int. J. Geo-Inf.* **2015**, *4*, 1055–1075. [CrossRef]
19. Ding, M.; Lyngbaek, K.; Zakhor, A. Automatic registration of aerial imagery with untextured 3D LiDAR models. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 8 June 2008; pp. 1–8. [CrossRef]
20. Awrangjeb, M.; Zhang, C.; Fraser, C.S. Automatic extraction of building roofs using LIDAR data and multispectral imagery. *ISPRS J. Photogramm. Remote Sens.* **2013**, *83*, 1–18. [CrossRef]
21. Huang, J.; Stoter, J.; Peters, R.; Nan, L. City3D: Large-Scale Building Reconstruction from Airborne LiDAR Point Clouds. *Remote Sens.* **2022**, *14*, 2254. [CrossRef]
22. Jovanović, D.; Milovanov, S.; Ruskovski, I.; Govedarica, M.; Sladić, D.; Radulović, A.; Pajić, V. Building virtual 3D city model for smart cities applications: A case study on campus area of the University of Novi Sad. *ISPRS Int. J. Geo-Inf.* **2020**, *9*, 476. [CrossRef]

23. Rumpler, M.; Irschara, A.; Wendel, A.; Bischof, H. Rapid 3D city model approximation from publicly available geographic data sources and georeferenced aerial images. In Proceedings of the Computer Vision Winter Workshop, Mala Nedelja, Slovenia, 1–3 February 2012; pp. 1–8. Available online: https://www.tugraz.at/fileadmin/user_upload/Institute/ICG/Images/team_fraundorfer/personal_pages/markus_rumpler/citymodeling_cvww2012.pdf (accessed on 1 January 2024).
24. Buyukdemircioglu, M.; Kocaman, S.; Isikdag, U. Semi-automatic 3D city model generation from large-format aerial images. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 339. [\[CrossRef\]](#)
25. Müller, P.; Zeng, G.; Wonka, P.; Van Gool, L. Image-based procedural modeling of facades. *ACM Trans. Graph.* **2007**, *26*, 85. [\[CrossRef\]](#)
26. Affara, L.; Nan, L.; Ghanem, B.; Wonka, P. Large scale asset extraction for urban images. In Proceedings of the European Conference on Computer Vision—ECCV 2016, Lecture Notes in Computer Science, 9907, Amsterdam, The Netherlands, 11–14 October 2016; pp. 437–452. [\[CrossRef\]](#)
27. Martinović, A.; Mathias, M.; Weissenberg, J.; Gool, L.V. A three-layered approach to facade parsing. In Proceedings of the European Conference on Computer Vision—ECCV 2012, Florence, Italy, 7–13 October 2012; pp. 416–429. [\[CrossRef\]](#)
28. Badrinarayanan, V.; Kendall, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [\[CrossRef\]](#) [\[PubMed\]](#)
29. Schmitz, M.; Mayer, H. A convolutional network for semantic facade segmentation and interpretation. *ISPRS Photogramm. Remote Sens. Spat.* **2016**, *41*, 709–715. [\[CrossRef\]](#)
30. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision—ECCV 2020, Glasgow, UK, 23–28 August 2020; pp. 213–229. [\[CrossRef\]](#)
31. Dore, C.; Murphy, M. Semi-automatic generation of as-built BIM façade geometry from laser and image data. *J. Inf. Technol. Constr.* **2014**, *19*, 20–46. Available online: <https://www.itcon.org/2014/2> (accessed on 1 January 2024).
32. Agarwal, S.; Furukawa, Y.; Snavely, N.; Simon, I.; Curless, B.; Seitz, S.M.; Szeliski, R. Building Rome in a day. *Commun. ACM* **2011**, *54*, 105–112. [\[CrossRef\]](#)
33. Snavely, N.; Seitz, S.M.; Szeliski, R. Photo tourism: Exploring photo collections in 3D. *ACM Trans. Graph.* **2006**, *25*, 835–846. [\[CrossRef\]](#)
34. Zhang, K.; Riegler, G.; Snavely, N.; Koltun, V. NeRF++: Analyzing and improving neural radiance fields. *arXiv* **2020**, arXiv:2010.07492. [\[CrossRef\]](#)
35. Tancik, M.; Casser, V.; Yan, X.; Pradhan, S.; Mildenhall, B.; Srinivasan, P.P.; Barron, J.T.; Kretzschmar, H. Block-NeRF: Scalable large scene neural view synthesis. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 8238–8248. [\[CrossRef\]](#)
36. Bódis-Szomorú, A.; Riemenschneider, H.; Van Gool, L. Efficient volumetric fusion of airborne and street-side data for urban reconstruction. In Proceedings of the 23rd International Conference on Pattern Recognition, Cancun, Mexico, 4–8 December 2016; pp. 3204–3209. [\[CrossRef\]](#)
37. Kelly, T.; Femiani, J.; Wonka, P.; Mitra, N.J. BigSUR: Large-scale structured urban reconstruction. *ACM Trans. Graph.* **2017**, *36*, 1–16. [\[CrossRef\]](#)
38. Hertzmann, A.; Jacobs, C.E.; Oliver, N.; Curless, B.; Salesin, D.H. Image analogies. In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 1 August 2021; pp. 327–340. [\[CrossRef\]](#)
39. Wei, L.Y.; Levoy, M. Fast texture synthesis using tree-structured vector quantization. In Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, New Orleans, LA, USA, 23–28 July 2000; pp. 479–488. [\[CrossRef\]](#)
40. Henzler, P.; Mitra, N.J.; Ritschel, T. Learning a neural 3D texture space from 2D exemplars. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 8356–8364. [\[CrossRef\]](#)
41. Vidanapathirana, M.; Wu, Q.; Furukawa, Y.; Chang, A.X.; Savva, M. Plan2Scene: Converting floorplans to 3D scenes. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 10733–10742. [\[CrossRef\]](#)
42. Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; Geiger, A. Occupancy networks: Learning 3D reconstruction in function space. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4460–4470. [\[CrossRef\]](#)
43. Ogawa, Y.; Oki, T.; Chen, S.; Sekimoto, Y. Joining street-view images and building footprint GIS data. In Proceedings of the 1st ACM SIGSPATIAL International Workshop on Searching and Mining Large Collections of Geospatial Data, Beijing, China, 8 November 2021; pp. 18–24. [\[CrossRef\]](#)
44. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969. [\[CrossRef\]](#)
45. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. In Proceedings of the European Conference on Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014; pp. 740–755. [\[CrossRef\]](#)
46. Sun, K.; Xiao, B.; Liu, D.; Wang, J. Deep high-resolution representation learning for human pose estimation. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5693–5703. [\[CrossRef\]](#)

47. Zhou, B.; Zhao, H.; Puig, X.; Fidler, S.; Barriuso, A.; Torralba, A. Scene parsing through ADE20K dataset. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 633–641. [\[CrossRef\]](#)
48. Zhou, B.; Zhao, H.; Puig, X.; Xiao, T.; Fidler, S.; Barriuso, A.; Torralba, A. Semantic understanding of scenes through the ADE20K dataset. *Int. J. Comput. Vis.* **2019**, *127*, 302–321. [\[CrossRef\]](#)
49. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the 2019 IEEE/CVF conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 658–666. [\[CrossRef\]](#)
50. Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; et al. Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Venice, Italy, 21–26 July 2017; pp. 105–114. [\[CrossRef\]](#)
51. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015. [\[CrossRef\]](#)
52. Ashikhmin, M. Synthesizing natural textures. In Proceedings of the 2001 Symposium on Interactive 3D Graphics, Chapel Hill, NC, USA, 26–29 March 2001; pp. 217–226. [\[CrossRef\]](#)
53. Efros, A.A.; Freeman, W.T. Image quilting for texture synthesis and transfer. In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 12–17 August 2001; pp. 341–346. [\[CrossRef\]](#)
54. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255. [\[CrossRef\]](#)
55. Arthur, D.; Vassilvitskii, S. k-means++: The advantages of careful seeding. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, LA, USA, 7–9 January 2006; pp. 1027–1035. Available online: <https://theory.stanford.edu/~sergei/papers/kMeansPP-soda.pdf> (accessed on 1 January 2024).
56. Fan, H.; Kong, G.; Zhang, C. An Interactive platform for low-cost 3D building modeling from VGI data using convolutional neural network. *Big Earth Data* **2021**, *5*, 49–65. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.