

## Article

# Developing a Framework for Data-Driven Generation of Building Information Modeling from Sketches: Enhancing Efficiency in Space Configuration and Building Performance Analysis

WoonSeong Jeong <sup>1</sup>, ByungChan Kong <sup>1</sup> and Sang-Guk Yum <sup>2,\*</sup>

<sup>1</sup> Department of Architectural Engineering, Chungbuk National University, Cheongju 28644, Republic of Korea; wsjeong@chungbuk.ac.kr (W.J.); skckwdk@chungbuk.ac.kr (B.K.)

<sup>2</sup> Department of Civil Engineering, Gangneung-Wonju National University, Gangneung 25457, Republic of Korea

\* Correspondence: skyeom0401@gwnu.ac.kr; Tel.: +82-33-640-2420

**Abstract:** The demand for compact housing is on the rise, driven by the need for floor plans that accommodate stakeholders' preferences. However, clients frequently struggle to convey their spatial needs to professionals, such as architects, due to a lack of means to present evidence, such as spatial configurations or cost projections. This study seeks to develop a methodology that translates sketched, data-driven spatial requirements into 3D building components within BIM (Building Information Modeling) to enhance spatial comprehension and offer building performance analysis, assisting in budget considerations during the initial design stages. The research methodology encompasses the formulation of a process model, its implementation, and subsequent validation. The process model outlines the data flow within the system and delineates necessary functionalities. Implementation includes the creation of systems and user interfaces for the integration of various components. Validation confirms the system's capability to automatically transform sketched spatial requirements into BIM model elements, such as walls, floors, and roofs, and to autonomously compute material and energy expenses based on the BIM model. This system enables clients to effectively generate 3D building components from sketches, aiding stakeholders in spatial understanding and building performance evaluation through the generated BIM models.

**Keywords:** building information modeling; deep learning; sketched data retrieval; building performance analysis; cost estimation



**Citation:** Jeong, W.; Kong, B.; Yum, S.-G. Developing a Framework for Data-Driven Generation of Building Information Modeling from Sketches: Enhancing Efficiency in Space Configuration and Building Performance Analysis. *Appl. Sci.* **2024**, *14*, 3013. <https://doi.org/10.3390/app14073013>

Academic Editor: Asterios Bakolas

Received: 2 March 2024

Revised: 22 March 2024

Accepted: 28 March 2024

Published: 3 April 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 incidence of single-person households is steadily increasing [1], and the discourse on providing compact housing to enhance residential satisfaction for single and two-person households continues to gain momentum in South Korea [2]. These residential preferences indicate a preference for personalized living spaces, a trend mirrored in the design of compact and micro-housing units [3]. Such housing types allow for the client's spatial requirements to be actively incorporated into the floor plan, operating within a more modest budget for construction and maintenance costs compared to detached houses [3]. However, clients encounter limitations in providing concrete evidence of their spatial needs, such as spatial configurations or cost estimates, prior to engaging in floor plan discussions with professionals, including architects. This indicates that considerable time and effort are expended by the clients to accurately convey their spatial requirements to stakeholders.

During the early design stages, non-professional clients, including homeowners (clients), often struggle to articulate their spatial needs—such as size and layout—into concrete building components and envision the final structure without a considerable

investment of time and effort, or the aid of architectural experts, such as architects. While floor plans, perspectives, and interior perspective images are provided to assist clients in understanding space, these limited resources necessitate more explicit materials for a comprehensive understanding of spatial arrangements and circulation [4]. Consequently, there are steady needs for creating an effective platform that accurately captures clients' spatial needs right from the conceptual and early design phases. This platform should facilitate a better grasp of spatial layouts, offering a clearer and more impactful alternative to 2D data using 3D visualizations [5].

Furthermore, relying on 2D information during the initial design phases has proven inefficient for generating construction cost estimates and for the analysis of building performance, including material qualities, physical attributes, and energy consumption [6–8]. Additionally, clients consistently seek a system that enables an efficient assessment of their spatial needs within budgetary limits. Therefore, it is essential to create a methodology that enables clients to visualize their spatial concepts clearly in 3D during the early design stages, as well as to use these 3D models to analyze building performance, thereby evaluating budget feasibility.

This study aims to develop a methodology that actualizes spatial requirements derived from sketches provided by clients, encompassing non-expert homeowners, into 3D architectural elements during the initial design phase. This enables a thorough examination of spatial visualization and facilitates an analysis of the associated construction and maintenance costs. The developed framework from the methodology automatically generates a Building Information Modeling (BIM) model from the spatial requirements (sketch information) expressed through a sketching tool. It extracts shape and material property information from the components of the BIM model, directly computing building performance analysis information such as material costs and electricity and gas usage charges. The 3D spaces created through the developed framework can support users in explicitly understanding the spatial arrangement, and the derived building performance analysis information can serve as evidence for decision-making support in budget management.

## 2. Literature Reviews

In the initial design phase, research supporting clients' decision-making on spatial configuration has been extensively conducted, and such decisions in the early stages of space arrangement significantly influence the final quality and cost estimation of a building [9]. However, studies on developing an environment that provides clients with explicit spatial understanding, construction cost estimation, and comprehensive building performance analysis at this stage are still limited.

Choi et al. and Jung et al. [10,11] provided an environment that enables the review of circulation, floor and elevation plans, along with construction costs during the initial design phase. However, clients continue to demand information for decision-making regarding budget appropriateness by reviewing not only construction costs from a financial perspective but also information on building maintenance costs through building performance analysis, such as energy consumption, at the early design stages.

Cho et al. [12] facilitated intuitive understanding of space for users by translating floor sketches into building components within a BIM model framework during the initial design phase. However, there is an additional need for an environment capable of generating the basis for decision-making support in budget management, such as construction costs and building performance analysis information.

Recent studies emphasize the need for research focused on efficiently generating preliminary design models in BIM. This effort aims to firmly establish the spatial and performance criteria of buildings as defined by stakeholders. The objective of this research is to facilitate the effective creation of Building Information Modeling (BIM) models from sketches supplied by clients. The developed framework automates the translation of the sketched information into precise architectural elements, such as walls, floors, and roofs, in a BIM model. The resultant BIM models not only precisely reflect the requirements of

the stakeholders—enhancing architects’ grasp of their intentions—but also incorporate material attributes. This incorporation is crucial for the preliminary assessment of building performance, including energy consumption and cost estimation of materials, via the BIM models. As a result, the framework is instrumental in aiding the decision-making process for spatial planning and construction, ensuring that the initial design decisions are in line with both functionality and budgetary constraints.

### 3. Developing a Framework for Data-Driven Generation of Building Information Modeling from Sketch Information

To address the limitations identified in existing research, this study develops a framework that efficiently generates BIM models based on floor sketches reflecting clients’ spatial requirements and effectively computes building performance analysis outcomes using the BIM models. Clients employ sketching tools to articulate spatial information, such as floor plan configurations and flooring shapes. This sketch information is utilized by the framework to automatically create 3D building components, including walls, floors, and roofs, within the BIM model, and the framework then automatically calculates material costs and annual electricity and gas energy expenses. Through the developed framework, stakeholders, including non-professional clients, can access the following functionalities during the early design phase.

- Providing spaces constructed from 3D building components, enabling a more intuitive and explicit visualization of space than would be possible with 2D drawings and interior images.
- Supplying a data model capable of efficiently conveying building performance analysis outcomes to stakeholders via the BIM model.
- Presenting stakeholders with a detailed overview of specific spatial arrangements, direct material expenditures, and results of energy performance analysis.

In this research, we delineated two distinct phases to develop a framework aimed at the automated generation of BIM models from sketch information, incorporating building performance analysis such as cost estimation and energy consumption evaluation: (1) development of a process model, (2) implementation of the framework, and (3) validation of the framework. The development of the process model involved identifying and defining the specific steps required for framework development, as well as articulating and defining the relationships between data inputs, processing, and outputs. To develop the process model, we identified the flow of data and established a series of steps comprising activities. In the framework implementation phase, the activities defined in the process model were realized through system and user interfaces. The implementation of the system interface involved coding the algorithms defined in the process model and defining methods for data exchange between disparate activities. The user interface was developed to enable users to interact with the system interface and operate the system effectively. The validation verifies whether the framework can automatically translate sketch information into a BIM model consisting of walls, floors, and roofs, as well as conduct building performance analysis.

#### 3.1. Development of a Process Model

The process model was designed to define the processes required for the development of the framework. It established a series of activities that describe the system’s functionalities and defined the information required for these activities, as well as the relationships between these pieces of information.

For the development of the process model, the International Organization for Standardization’s Integration Definition for Function Modeling 0 (IDEF0) technique was utilized [13]. To implement the process model, the standardized IDEF0 method was employed to depict the relationships between activities and between disparate activities using diagrams and textual descriptions involving boxes and arrows. The specific representation techniques of the IDEF0 modeling method employed in this study are as follows.

- Activity: Represented textually within a box.

- Input: Denoted by arrows entering the box from the left.
- Output: Indicated by arrows exiting the box to the right.
- Condition: Expressed by arrows coming into the box from the top.
- Mechanism: Shown by arrows entering the box from the bottom.
- Sequence of Activities: Depicted by connecting diverse boxes with arrows.

Figure 1 illustrates the process model developed through the IDEF0 modeling technique. The process model is composed of three stages: (1) Extracting vector information (Activity 1, A1); (2) Creating a BIM model (Activity 2, A2); (3) Performing building performance analysis (Activity 3, A3).

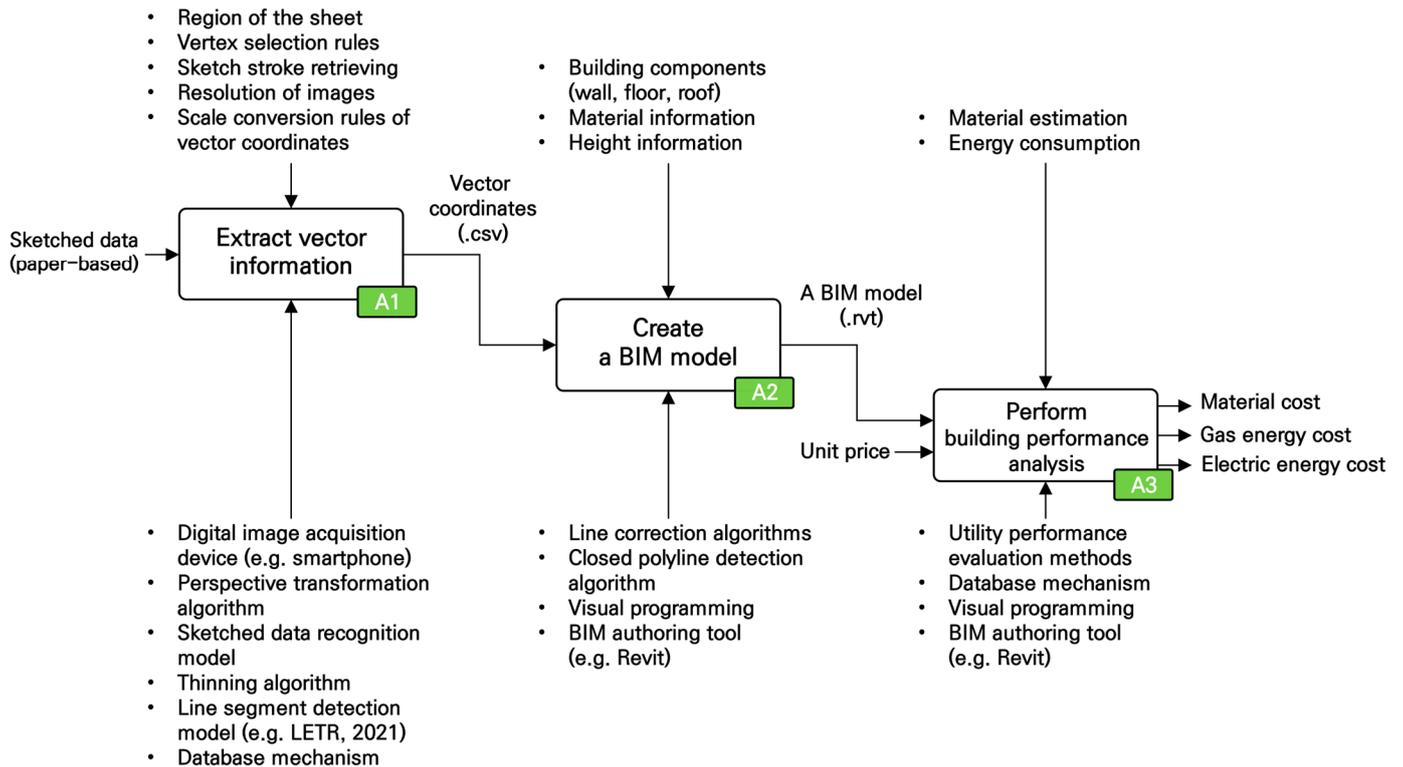


Figure 1. The process model outlines the comprehensive workflow of the framework.

### 3.1.1. Extracting the Vector Information

The process of extracting vector information, as depicted in Figure 1’s Activity 1, utilizes sketch information provided by the clients as input and outputs vector information used for generating building components, such as walls from the sketch information. Given that the sketch information is in an analog format using paper and pen, it was converted into a digital image format through digital image acquisition portable devices, such as smartphones. Smartphones were employed for their ease of converting sketch information into a digital format, compared to more complex devices, such as scanning equipment and Sensors following the LiDAR technology.

The digitally converted sketch information (sketch image) includes not only the sketch strokes that represent the centerline information of walls and the scale bars indicating scale information but also unnecessary elements, such as shadows, reflections, and the desk surface. To efficiently extract vector information used for creating building components from the sketch information, image preprocessing was conducted to remove these unnecessary details from the sketch image. The image preprocessing utilized techniques including (1) a perspective transformation algorithm, (2) a sketch information recognition model, and (3) a thinning algorithm.

Through the perspective transformation algorithm, the selected area within the sketch image was transformed into a predefined rectangular shape, effectively removing unneces-

sary information, such as the desk surface, existing outside the paper boundary. To select the internal area of the paper for perspective transformation within the sketch image, a set of rules for choosing the paper's outer corner points was defined. The sketch information recognition model identified sketch strokes within the sketch image and eliminated unnecessary elements such as shadows and reflections inside the paper, excluding the sketch strokes. The implementation of the sketch information recognition model utilized the Pytorch 1.8.2+cu111 [14] deep learning framework, and a thinning algorithm was applied to reduce the thickness of the sketch strokes to a single pixel in the sketch image. Pytorch, a scientific computing and deep learning library, accelerates computations via GPUs, potentially speeding up operations by up to 50 times compared to CPUs. It also offers models for numerical optimization, streamlining the development of deep learning models. This preprocessing step was performed to efficiently extract the centerline information of walls from the sketch strokes. The preprocessed sketch image was then separated into images of the wall centerlines and scale bars, each saved as individual image files. The centerline image of the wall was created to extract the central line information from individual sketch strokes, and the scale bar image was generated to adjust the centerline information to the actual lengths intended by the user, based on which dimension information was calculated.

To extract the centerline information used for generating building components from the centerline images of walls, a deep learning-based line detection model, LETR [15], was employed. The LETR model detected the centerline information (vector information) from individual sketch strokes (pixel information) in the input centerline images of walls. The extracted centerline information was converted into a Scalable Vector Graphics (SVG) file format, and the vertical and horizontal resolution of the SVG files was adjusted to transform the lengths to those actually intended by the client. The specific values for adjusting the image resolution were calculated using the predefined scale information (1:50) and the width of the scale bar in the scale bar image (34 pixels). The length-transformed centerline information was defined within a database mechanism to be convertible into a Comma Separated Values (CSV) file format, facilitating information exchange with BIM authoring tools such as Revit [16].

### 3.1.2. Creating a BIM Model

The creation of a BIM model, depicted in Figure 1's Activity 2, involves using vector information, specifically centerline data converted into a CSV format, as input. This process creates a BIM model that accurately represents building components, including walls, floors, and roofs. A BIM authoring tool and visual programming techniques were employed to automatically create 3D building components within the BIM model. When generating building component objects in BIM models using Revit, overlapping errors between different objects at the same location can prevent automatic creation. To automate the creation of building component objects in BIM models, a line correction algorithm was developed. This algorithm adjusts the lengths of input centerlines and removes lines with similar positions, directions, and lengths. Additionally, the line correction algorithm was used to adjust digital floor plans from various sketching techniques used by different users. The corrected centerline information, along with attribute information, such as material (concrete) and location (first floor), was utilized to create building component objects such as walls. An algorithm to extract the outermost lines from the centerline information was defined for generating polygons used in creating building components such as floors and roofs. The extracted outermost lines, along with attributes such as material and floor information, were employed to generate floor and roof objects.

### 3.1.3. Performing Building Performance Analysis

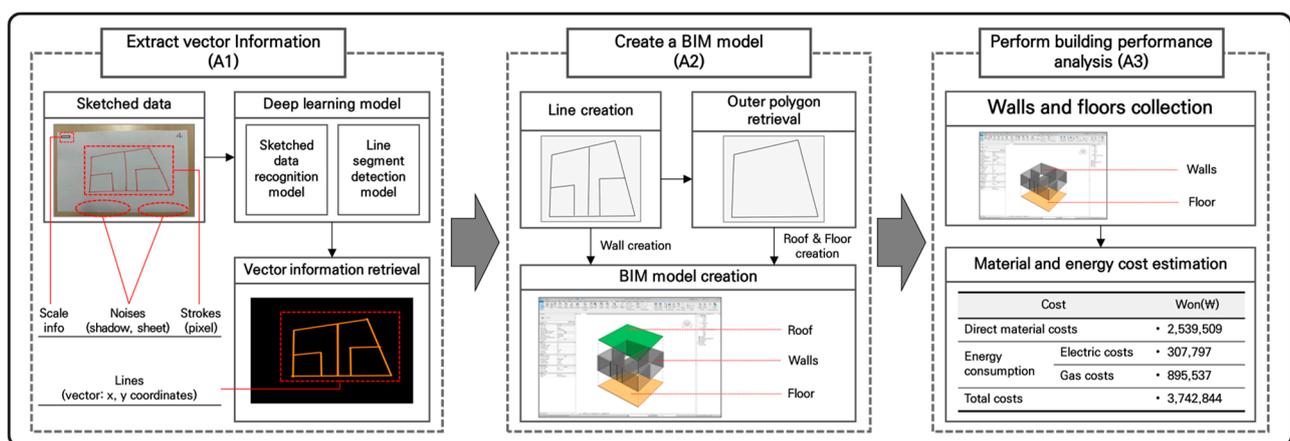
The procedure for conducting building performance evaluations, as depicted in Figure 1's Activity 3, leverages the BIM model and material cost information as inputs. This process calculates building performance metrics, encompassing direct material expenses and annual charges for electricity and gas energy. In this research, building performance is

defined in terms of construction costs and energy consumption, which can be estimated from the BIM model at the conceptual design stage. Construction costs are defined as direct material costs, and energy consumption is defined as annual charges for electricity and gas energy use. To calculate these building performance analysis data, algorithms were employed that automatically extract attribute information (volume, area) from objects in the BIM model and utilize BIM authoring tools and visual programming techniques to run these algorithms. An algorithm was defined to calculate direct material costs by extracting the total volume of wall objects from the BIM model, summing the volume of the floors, and applying the unit volume price of materials (concrete). To calculate energy charges, an algorithm was defined that extracts the area of floor objects from the BIM model and applies the unit area price for energy use. The unit area price for gas energy use was set at KRW 17,201 which is the currency code for the South Korean won (₩), and for electricity use at KRW 5912, based on energy consumption statistics from the Ministry of Land, Infrastructure, and Transport in Korea. The calculated building analysis information was defined to be convertible into a CSV file format through a database mechanism.

### 3.2. Implementation of the Framework

Through the process model, the input and output information for each stage of the framework was defined, and based on the individual activities of the process model, both the system interface and user interface were implemented. In implementing the system interface, every algorithm specified in the process model was executed, and a methodology for integrating data across different systems was formulated.

The implementation of the framework utilized the Python programming language [17] and the OpenCV computer vision library [18]. For the implementation of the sketch information recognition model in Activity 1 (A1), as shown in Figure 1, the Pytorch v.1.8.2+cu111 deep learning framework was employed. To implement Activities 2 (A2) and 3 (A3), the BIM authoring tool Revit 2024 [16] and the visual programming tool Dynamo [19] were used. Figure 2 illustrates the process through which the framework creates a BIM model based on sketch information provided by the user and calculates building performance analysis data from the created BIM model.



**Figure 2.** The workflow of the framework, which transforms sketched data into a BIM model and conducts building performance evaluations.

#### 3.2.1. Implementation of Extracting Vector Information System

The vector information extraction system implemented the process associated with Figure 2's Activity A1. A system for extracting vector information from sketch data, used to create wall, floor, and roof objects in a BIM model, was implemented using the Python programming language and the Pytorch deep learning framework. Table 1 displays the input data for the vector information extraction system and the data output through the applied mechanisms.

Table 1. Description of the vector data retrieval system.

Step Process	(1) Remove Noises from Outside in the Sheet	(2) Remove Noises from Inside in the Sheet	(3) Retrieve Center Lines	(4) Split the Image	(5) Retrieve Vector Data
	Sketch image	Pre-processed image	Wall image	Pre-processed image	Center line image
Input					
	Perspective transformation algorithm	Sketched data recognition model	Thinning algorithm	Image manipulation	Line segment detection model
Method /Mechanism					
	Prepared image	Wall image	Pre-processed im-age	① Scale-bar image ② Center line im-age	Vector data
Output					

A perspective transformation application was developed to eliminate unnecessary information, such as the desk surface outside the paper boundary in sketch images, that is not part of the sketch strokes. This application utilized perspective transformation functions and functions for accessing and editing image files provided by the OpenCV library. The perspective transformation application allows users to select four corner points of the paper boundary in a clockwise direction within the sketch image. Following selection, the area defined by these corner points is saved as an image with a predefined resolution (width: 2016 pixels, height: 1512 pixels). It also offers functionality for users to select the area (four corner points) for perspective transformation application through a left mouse click. Table 1’s column, “Remove noises from outside the sheet”, illustrates the removal of unnecessary information from outside the paper boundary in the sketch image through the perspective transformation application.

A sketch information recognition model was developed to eliminate extraneous information, such as shadows, reflections, and paper texture, within the sketch image that is not part of the sketch strokes. The development of the sketch information recognition model utilized the skip connection, encoding, and decoding architecture of Unet++ [20], with individual Convolution units of Unet++ implemented as Residual units from ResUnet [21]. The sketch information recognition model was developed using ResUnet architecture, originally designed for road detection in remote sensing images due to its effective binary classification with minimal training data. This study leveraged the morphological parallels between roads in remote sensing and sketch strokes, alongside the similar binary classification challenge of differentiating sketch strokes from non-stroke elements. Thus, a deep learning model based on ResUnet was created to accurately identify sketch strokes in images. The architecture of Unet++ was employed to accurately identify pixels corre-

sponding to sketch strokes in sketch images, while the Residual units of ResUnet were used to facilitate model training with a limited amount of training data. The combination of both models allowed for the implementation of the sketch information recognition model capable of high-accuracy recognition of sketch information with limited training data. The model was implemented with an encoding and decoding structure of Unet++ composed of 15 Residual units. Each individual Residual unit in the encoding structure was progressively connected to each Residual unit in the decoding structure through skip connections. Each Residual unit performed operations combining a  $3 \times 3$  convolution layer, batch normalization, and ReLU activation function, followed by an addition operation with the initial input variable,  $x$ . The Pytorch deep learning framework and the Python programming language were used to implement the sketch information recognition model. For training the model, a dataset of 50 images with a resolution of 224 pixels in width and height was created, consisting of input images and mask images. The input images used were perspective-transformed sketch images, and the mask images were used to represent images with sketch strokes. Choosing the right loss and optimization functions is essential for a deep learning model to effectively optimize parameters and improve accuracy. For a sketch recognition model, which aims to classify pixels in a sketch as either strokes or not, the Binary Cross Entropy loss function was selected for its relevance to binary classification problems. This function measures the model's output against the true segmentation. Given the non-linear and complex nature of sketch information, which includes variations in spatial requirements, pen thickness, and the effect of hand tremor, the Adam [22] optimizer was used to efficiently adjust the model's parameters. Both components were implemented via Pytorch. The number of training epochs was set to 150, and the batch size was set to 16. Table 1, column 2 "Remove noises from inside the sheet", illustrates the model's ability to recognize sketch strokes (white pixels) and classify non-stroke information (black pixels) in the sketch image.

A wall shape representation image (wall image) was generated using the sketch information recognition model from sketch images. To convert the thickness of sketch strokes in the wall image to a single pixel, a thinning algorithm [23] was implemented using the OpenCV library. This algorithm removed pixels not central to individual sketch strokes in the wall image. An algorithm was developed to save the thinned wall image into designated areas within the image as separate files for the scale bar image and the wall's centerline image. Table 1, in the column "Retrieve center lines", illustrates the application of the thinning algorithm to convert the thickness of sketch strokes to a single pixel, and the "Split the image" column shows the process of separating the wall image into the scale bar image and the wall's centerline image.

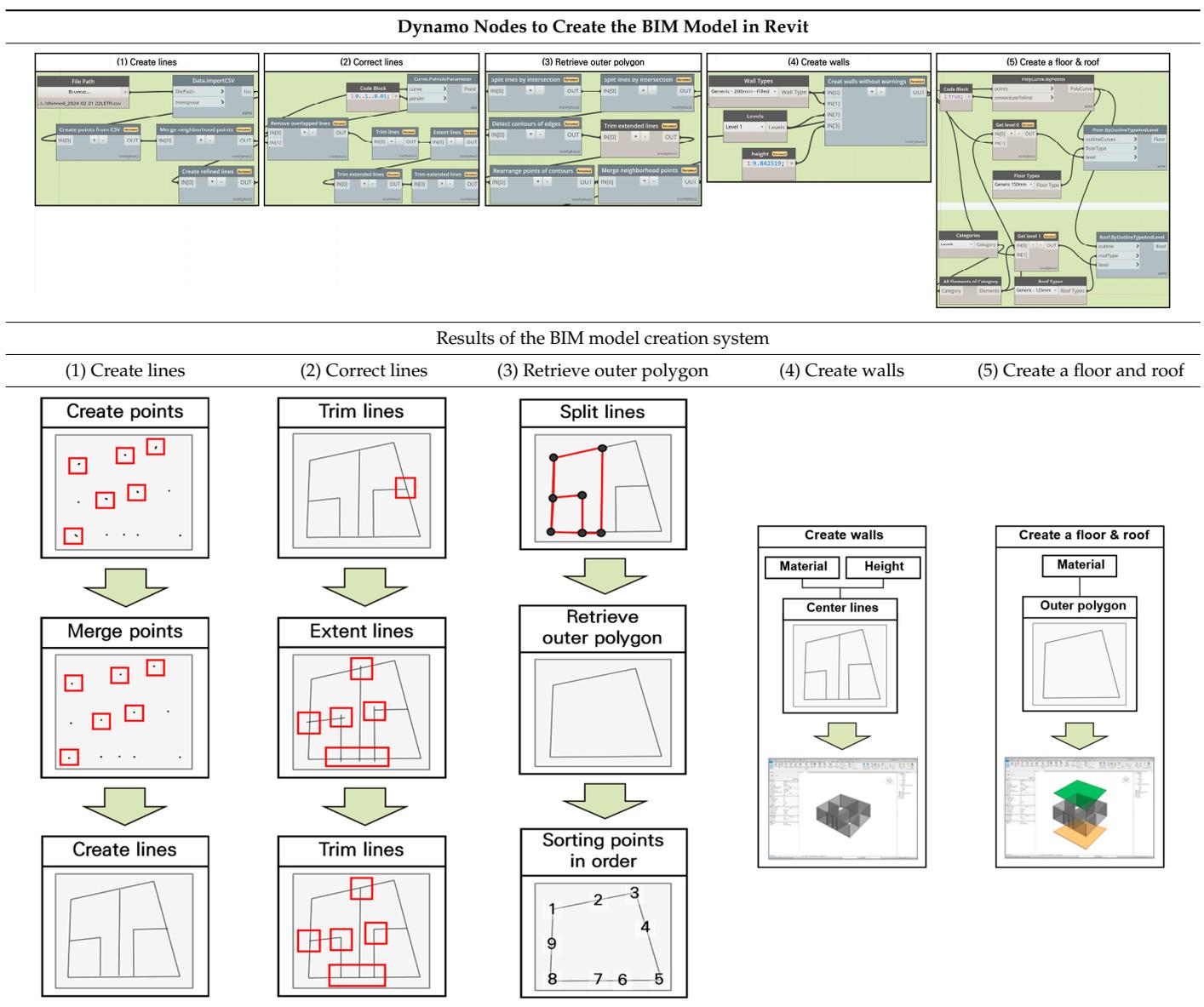
For extracting the central line information (centerlines) of individual walls from the wall's centerline image, the LETR model by [15] was adopted. The LETR model, upon inputting an image, detects lines within the image and outputs the  $x, y$  coordinates of the start and end points of individual lines. The centerline extraction mechanism utilizing the LETR model was set in two stages: (1) extracting morphological features of neighboring pixels in the wall's centerline image, and (2) selecting detailed line objects in the wall's centerline image. The wall's centerline image was input into the LETR model with a resolution of 256 pixels in width and height, and Table 1's column "Retrieve vector data" displays the extraction of centerlines from the wall's centerline image using the LETR model. An algorithm was developed to save the centerlines at a resolution of 512 pixels in width and 362 pixels in height in a Scalable Vector Graphics (SVG) file format, adjusting the length of the centerlines to the length intended by the user, utilizing the OpenCV library. The adjustment of the SVG file's vertical and horizontal resolution and the length of the centerlines were calculated using the width of the white pixels in the scale bar image (34 pixels), the physical length of the scale bar represented on the paper (2 cm), and a predefined scale ratio (1/50). An algorithm was implemented in the Python programming language to extract the  $(x, y)$  coordinates of the start and end points of the centerlines from

the resolution-adjusted SVG file and to save them in the Comma Separated Values (CSV) file format.

### 3.2.2. Implementation of Creating a BIM Model System

The BIM model creation system implemented the process associated with Figure 2’s Activity A2. A system that utilizes CSV files, containing information on centerlines (lines), as the initial input data to create a BIM model representing walls, floors, and roofs in the BIM authoring tool Revit, was implemented using the Python language and the visual programming tool Dynamo in Table 2. The BIM model generation system is structured into five detailed steps: (1) generating points and lines from the CSV file, (2) applying a line correction algorithm, (3) applying an outermost line extraction algorithm, (4) creating wall objects, and (5) creating roof and floor objects.

Table 2. Development of the BIM model creation system.



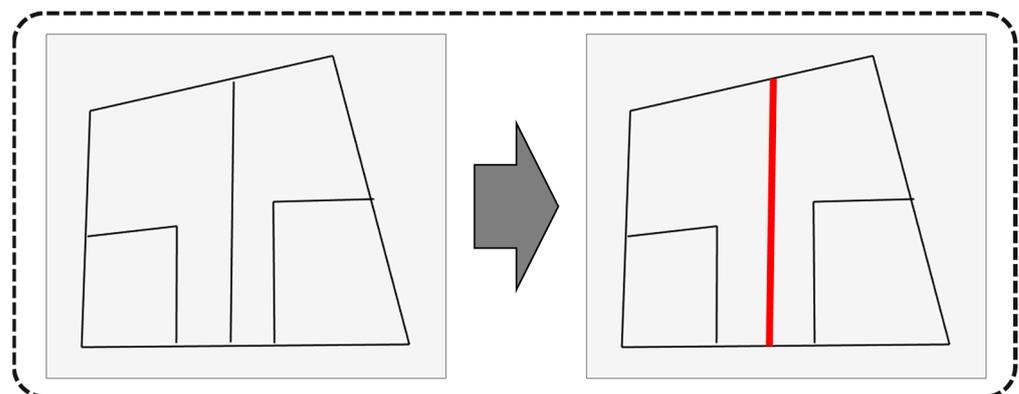
The process of generating points and lines from the CSV file encompasses three steps: (1) generating points, (2) integrating neighboring points, and (3) creating lines. The algorithm for generating points employs a CSV file, which details the point coordinates of lines, as input data to output the  $x, y$  coordinates of the lines’ start and end points. These

output  $x, y$  coordinates are then set as input data for the Dynamo nodes that create points in Revit, facilitating the output of the start and end points of the lines. In Table 2, the “Create points” under the “Create lines” column illustrates the start and end points of the centerlines created in Revit, where the generated points, as shown in the red box in “Create points” under the “Create lines” column are scattered instead of being at the same location. The algorithm for aligning neighboring points takes the start and end points of lines as input data, generates a circle of a fixed radius (0.2 m) around each point, and eliminates all points on each circle except for one. “Merge points” under the “Create lines” column in Table 2 displays the integration of points into a single point. The algorithm for creating lines sets the start and end points as input data for the Dynamo nodes (nodes that create points) and implements the output of lines, with “Create lines” under the “Create lines” column in Table 2 showing the lines created in Revit.

The line correction algorithm comprises three stages: (1) removing and trimming lines, (2) extending lines, and (3) trimming lines. The removing and trimming stage employ lines as input data to eliminate all lines that overlap, share similar lengths and angles, except for the longer lines, and to remove segments that pass through the intersection of two different lines. To define lines that overlap and share similar lengths and angles for the first stage, lines from the input data that meet the following three criteria were selected.

- The angle difference between two lines is within a specific threshold (10 degrees).
- The two lines intersect each other.
- For two lines meeting the above conditions, numerous points (100) are generated on the longer line, so lines that intersect with a shorter line at a specific number of points (40 or more).

Figure 3 illustrates the process of selecting lines that are similar in length and angle and overlap each other, based on the above conditions. The duplicated lines identified are automatically removed, followed by a process that eliminates unnecessarily extended portions based on the lines’ intersection points. In Table 2, “Remove overlapped lines” and “Trim lines” nodes under the “Correct lines” column depicts the trimming performed to remove these unnecessarily extended lines.



**Figure 3.** The process of the identified overlapping lines.

The extending lines phase was implemented to use lines as input data, targeting different lines that do not intersect within a specific distance (0.1 m), to output lines whose lengths have been extended. In Table 2, “Extend lines” under the “Correct lines” column displays the process of extending the length (0.3 m) of lines based on their individual starting and ending points, which do not intersect with each other. The extended lines intersect with other lines, necessitating the use of the previously implemented “Remove and trim lines” step in Table 2 under the “Correct lines” column to eliminate portions that pass-through intersection points. “Trim lines” in Table 2 under the “Correct lines” column shows the trimming performed on the extended lines.

The outermost line extraction phase is organized into three detailed steps: (1) dividing lines, (2) extracting the outermost lines, and (3) sequentially arranging the points. The dividing lines stage utilizes lines that have been adjusted by the line-correction algorithm as input data. It is implemented to split a line into two at an intersection point when an intersection occurs on the line. "Split lines" under the "Retrieve outer polygon" column in Table 2 illustrates the division of each line at intersection points. The second detailed step, extracting the outermost lines, uses the previously divided lines based on intersection points as input data to output the lines located furthest out. "Retrieve outer polygon" in Table 2 displays the extracted outermost lines. To utilize the extracted outermost lines for creating floor and roof objects, they were converted into a polyline format. To convert into the polyline format, an algorithm was first implemented to arrange the points that make up the outermost lines in clockwise or counterclockwise order. "Sorting points in order" under the "Retrieve outer polygon" column in Table 2 shows the points making up the outermost lines arranged in a clockwise direction. Secondly, the points arranged in a clockwise direction were used as input data for the Dynamo node that creates polylines, and the converted outermost polyline was outputted.

The wall object creation phase utilized the output from "Correct lines" in Table 2 (corrected lines) as the centerlines for walls. Additional input data, such as material information (Generic 200 mm) and wall height (3 m), were used to generate wall objects in Revit. "Create walls" in Table 2 illustrates the creation of wall objects in Revit.

The floor and roof object creation phase used the output from "Retrieve outer polygon" in Table 2 (outermost lines) as contours for the floor and roof objects. Additional input data, including material information (Floor: Generic 150 mm, Roof: Generic 125 mm), were utilized to generate floor and roof objects in Revit. "Create a floor and roof" in Table 2 displays the generation of floor and roof objects in Revit.

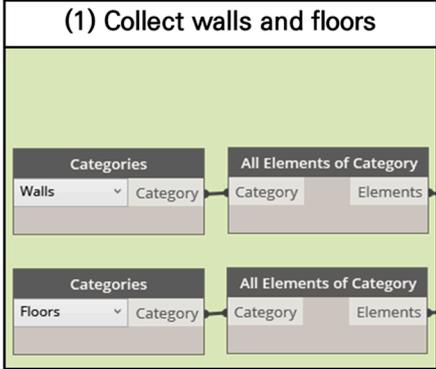
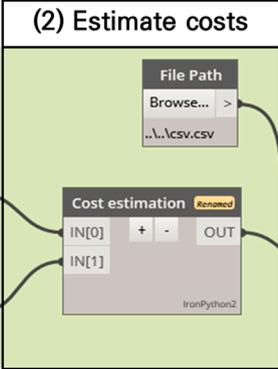
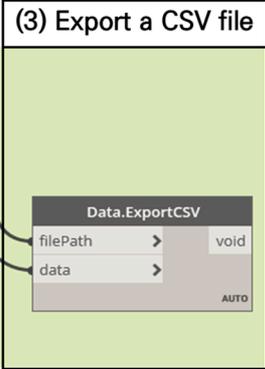
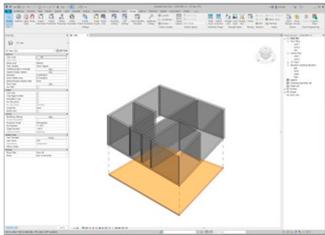
### 3.2.3. Implementation of Performing Building Performance Analysis System

The building performance analysis system implemented the process associated with Figure 2's Activity A3. The system was developed that utilizes the BIM model and unit price information for building performance analysis as input data to output direct material costs and electricity and gas energy charges. Table 3 illustrates the implementation of the building performance analysis information extraction system using Dynamo nodes. The system for extracting building performance analysis information is structured into three detailed steps: (1) collecting wall and floor objects, (2) calculating building performance analysis information, and (3) exporting to CSV files. The stage of collecting wall and floor objects employs the BIM model as input data to output wall and floor objects from the building components of the BIM model. The column "Collect walls and floors" in Table 3 shows the extraction of wall and floor objects from the BIM model.

The building performance analysis information extraction phase utilizes wall and floor objects along with unit price information as input data to calculate direct material costs and annual electricity and gas energy charges. An algorithm was developed to access the property information of wall and floor objects to extract volume information and perform a multiplication operation with the unit volume price of concrete (KRW 70,000) to calculate direct material costs. To calculate annual electricity energy charges, an algorithm was implemented that accesses the area information from the floor object properties and multiplies it by the annual electricity energy price per unit area (KRW 5912). The annual electricity energy price was defined based on the statistics of energy consumption for single-family homes from the Ministry of Land, Infrastructure, and Transport in Korea. For calculating annual gas energy charges, an algorithm was developed that extracts area information from floor object properties and multiplies it by the annual gas energy price per unit area (KRW 17,201), which was defined based on the gas billing method from SEOUL CITY GAS. Table 3's "Estimate costs" column displays the formulas used for the extraction of building performance analysis information. The total cost of building performance

analysis information was calculated by summing the direct material costs and annual electricity and gas energy charges.

**Table 3.** Development of the building performance estimation system utilizing dynamo.

Dynamo Nodes to Perform the Building Performance Analysis								
(1) Collect walls and floors	(2) Estimate costs	(3) Export a CSV file						
								
Description and results of the building performance analysis system								
(1) Collect walls and floors	(2) Estimate costs	(3) Export a CSV file						
	<table border="1"> <tr> <td style="text-align: center;">Material costs</td> <td style="text-align: center;">(Volume of Walls + Volume of Floors) × 70,000</td> </tr> <tr> <td style="text-align: center;">Electric costs</td> <td style="text-align: center;">Floor area × 5912</td> </tr> <tr> <td style="text-align: center;">Gas costs</td> <td style="text-align: center;">Floor area × 17,201</td> </tr> </table>	Material costs	(Volume of Walls + Volume of Floors) × 70,000	Electric costs	Floor area × 5912	Gas costs	Floor area × 17,201	
Material costs	(Volume of Walls + Volume of Floors) × 70,000							
Electric costs	Floor area × 5912							
Gas costs	Floor area × 17,201							

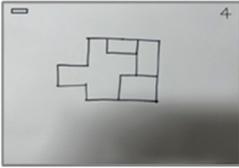
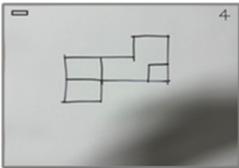
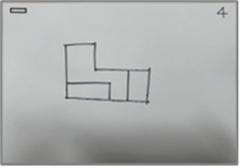
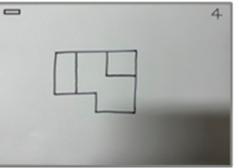
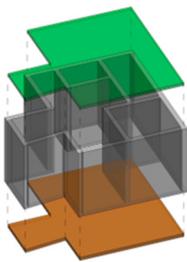
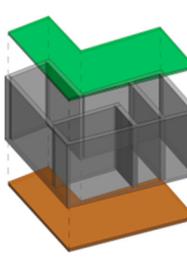
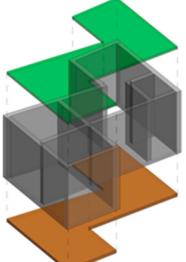
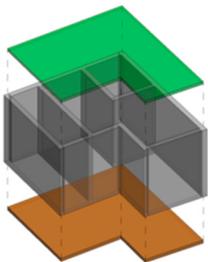
The exporting to CSV file phase was implemented using the calculated building performance analysis information and the path where the CSV file will be saved as input data, ensuring that the CSV file (containing building performance analysis information) is created at the specified path. In Table 3, “Export a CSV file” illustrates the execution of the created CSV file.

### 3.3. Validation

For the verification phase, the framework’s ability to automatically generate a BIM model using sketch information as input data was assessed, along with its capability to automatically produce building performance analysis information from the created BIM model. The validation method involved inputting various sketch information examples into the developed framework and first verifying whether wall, roof, and floor objects in the BIM model could be automatically created without errors. Additionally, a second verification step was established to determine whether direct material costs, and annual electricity and gas energy charges could be automatically calculated based on the generated BIM model. Table 4 displays the results of applying the framework to four sketch information examples, describing the created BIM models and the calculated building performance analysis information. Through Table 4, it was confirmed that each piece of sketch information (Input row) was transformed into wall, floor, and roof objects in the BIM model (Output row) using the developed framework. In the Direct material costs row of Table 4, it was verified that direct material costs were calculated by extracting volume information from the wall and floor objects of the BIM model. Similarly, in the Energy costs row, annual electricity and gas energy charges were calculated by extracting area information from the floor objects of the BIM model. Consequently, it was verified that the developed framework can

automatically generate a BIM model from sketch information and automatically produce building performance analysis information from the generated BIM model.

**Table 4.** Validation using test cases.

CASE	CASE 1	CASE 2	CASE 3	CASE 4
Input				
Output				
Direct material cost estimation (₩)	1,768,545	1,600,841	1,495,587	1,447,745
Energy cost estimation (₩)	Electric costs (₩)	156,422	129,365	137,616
	Gas costs (₩)	455,111	376,389	400,396
Total costs (₩)	2,380,080	2,106,595	1,944,831	1,985,758

#### 4. Conclusions and Discussion

This research developed a framework to support efficient spatial visualization and comprehensive decision-making based on building performance analysis within the BIM context during the initial design phase. The developed framework automatically transforms clients' spatial requirements, expressed through sketches, into BIM-based building component objects and effectively calculates building performance analysis information such as direct material costs and energy usage charges. Through the verification phase, it was confirmed that the framework automatically translates sketch information into wall, floor, and roof objects in a BIM model, and it was validated that the framework automatically calculates direct material costs, as well as annual electricity and gas energy charges from the BIM model.

The developed framework allows for the explicit visualization of spatial configurations from client-defined plan sketches through a BIM model, effectively providing clients with information on space size, form, and location during the initial design phase. Moreover, by efficiently calculating and providing direct material costs and annual electricity and gas energy charges from the generated BIM model, it serves as valuable evidence for budgetary decision-making at this stage. The generated BIM model can be utilized as a data model to efficiently convey explicit spatial requirements to professionals such as architects, serving as a foundational model for preliminary design.

Furthermore, the research aims to refine the framework by conducting case studies that demonstrate the creation of BIM models for multi-story buildings. Additionally, the study will explore enhancing the understanding of spatial configurations through experiences in virtual 3D environments created with BIM models, incorporating elements such as furniture and openings, including windows and doors. This approach utilizes the framework developed in this research, facilitating the comprehension of space based on sketch information provided directly by clients.

A more fundamental investigation into the potential errors within the BIM model creation system was conducted. The possibility of errors arising from translating sketched information into BIM components is noted. To investigate the mentioned error rate, six test

cases were executed. It was confirmed that the average error occurring when translating sketched information into components of a BIM model is 1.36%. This error rate is attributed to several factors: the performance limitations of the LETR model, a line detection model on an object level; distortions in line thickness resulting from the line refinement algorithm developed in this study; and length discrepancies arising from the wall join function within the Revit application. Future research will aim to reduce the error rate and improve the accuracy of the framework by conducting additional training of the LETR model and refining the line refinement algorithm.

While the framework has been implemented to calculate direct material costs and annual electricity and gas energy charges based on provided sketch information, further research is required to enhance the framework with accurate calculation and simulation techniques for more specific decision support in spatial detailing by providing additional building performance analysis information. For instance, the system will be further developed to allow stakeholders to select material information set as default values for building performance evaluation and to input building height information, offering detailed building performance analysis results instead of approximate analyses based on unit area.

Additionally, future research will address the general design process (creating mass before generating walls through spatial division) by enhancing the functionality to determine whether sketch information represents mass information or wall centerlines.

**Author Contributions:** Conceptualization, writing, visualization, funding acquisition, and methodology—W.J.; original draft writing, application development, and formal analysis—B.K.; writing review and supervision—S.-G.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2020R111A3052594).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are available upon request from the corresponding author. The data are not publicly available due to privacy.

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

## References

1. Park, M.S.; Lee, J.C. Housing Policy Responding to One-person Households Increase. *Krihs Policy Brief* **2018**, *665*, 1–8.
2. Huh, K.J.; Min, H.K. The Effects of Housing Cost Burden on Housing Choice: Differences by Household Type. *SH Urban Res. Insight* **2021**, *11*, 17–40. [[CrossRef](#)]
3. Jo, Y.J.; Yang, E.J.; Kim, K. A Study on the Spatial Composition of Urban Small-Housing for 1·2 People Household—Focused on domestic and Japanese narrow housing cases. *Bull. Korean Soc. Basic Des. Art* **2019**, *20*, 489–504.
4. Kim, J.S.; Lee, J.K. An Approach to the Graph-based Representation and Analysis of Building Circulation using BIM—MRP Graph Structure as an Extension of UCN. *Korean J. Constr. Eng. Manag.* **2015**, *16*, 3–11.
5. Di Stefano, F.; Gorreja, A.; Malinverni, E.S.; Mariotti, C. Knowledge Modeling for heritage Conservation process: From Survey to Hbim Implementation. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2020**, *44*, 19–26. [[CrossRef](#)]
6. Rezaei, F.; Bulle, C.; Lesage, P. Integrating building information modeling and life cycle assessment in the early and detailed building design stages. *Build. Environ.* **2019**, *153*, 158–167. [[CrossRef](#)]
7. Moon, H.J.; Kim, S.K. Development of a BIM based Energy Performance Evaluation Method for Buildings with a Twisted Shape in Early Design Stage. *J. Archit. Inst. Korea Plan. Des.* **2012**, *28*, 289–296.
8. Atazadeh, B.; Kalantari, M.; Rajabifard, A.; Ho, S.; Champion, T. Extending a BIM-based data model to support 3D digital management of complex ownership spaces. *Int. J. Geogr. Inf. Sci.* **2017**, *31*, 499–522. [[CrossRef](#)]
9. Gao, H.; Koch, C.; Wu, Y. Building information modelling based building energy modelling: A review. *Appl. Energ.* **2019**, *238*, 320–343. [[CrossRef](#)]
10. Choi, S.Y.; Choi, J.W.; Kim, J.H.; Kim, J.J. A Study on the Development of a BIM-based Spatial Planning Simulation System for Architectural Planning Stage Support. *J. KIBIM* **2011**, *1*, 19–23.
11. Jung, S.; Park, M.; Lee, H.S.; Yoon, I. Cost Estimation of Case-Based Reasoning Using Hybrid Genetic Algorithm-Focusing on Local Search Method Using Correlation Analysis. *Korean J. Constr. Eng. Manag.* **2020**, *21*, 50–60.
12. Cho, D.G.; Lee, J.K. Training Floorplan Sketches and Applying to the Spatial Design—Focused on the Development of Automated BIM Modeling module from Floor Plan Sketches in the Early Stage of Design. *J. Korea Institute Spat.* **2021**, *16*, 365–374.

13. National Institute of Standards and Technology. *FIPS Publication 183: Integration Definition of Function Modeling (IDEF0)*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 1993; Volume 128.
14. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, *2019*, 1–12.
15. Xu, Y.; Xu, W.; Cheung, D.; Tu, Z. Line segment detection using transformers without edges. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4257–4266.
16. Autodesk Revit Software | Get Prices & Buy Official Revit 2024. Available online: <https://www.autodesk.eu/products/revit/overview> (accessed on 2 March 2024).
17. Van Rossum, G.; Drake, F.L. *Python 3 Reference Manual*; Python Software Found: Scotts Valley, CA, USA, 2009.
18. Bradski, G. The openCV library. *Dr Dobbs J. Softw. Tools Prof. Program.* **2000**, *25*, 120–123.
19. Dynamo Studio | Computational BIM Design Software | Autodesk. Available online: <https://www.autodesk.com/products/dynamo-studio/overview> (accessed on 2 March 2024).
20. Zhou, Z.; Rahman Siddiquee, M.M.; Tajbakhsh, N.; Liang, J. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*; Stoyanov, D., Taylor, Z., Carneiro, G., Syeda-Mahmood, T., Martel, A., Maier-Hein, L., Tavares, J.M.R.S., Bradley, A., Papa, J.P., Belagiannis, V., et al., Eds.; Springer International Publishing: Berlin/Heidelberg, Germany, 2018; Volume 11045, pp. 3–11.
21. Zhang, Z.; Liu, Q.; Wang, Y. Road extraction by deep residual u-net. *IEEE Geosci. Remote Sens. Lett.* **2018**, *15*, 749–753. [[CrossRef](#)]
22. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the ICLR, San Diego, CA, USA, 7–9 May 2015.
23. Zhang, T.Y.; Suen, C.Y. A fast parallel algorithm for thinning digital patterns. *Commun. ACM* **1984**, *27*, 236–239. [[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.