Virtual Environment for Aerospace Simulation and AI Data: Focused on Automatic Building Generation

Goeun Yeo, Mungi Kim, Youngjun Sim, Dahyoung Jeon
SIM2REAL
Daejeon, South Korea
edisin67@gmail.com, kim.mungi@sim2real.co.kr, slkafsim@gmail.com, jeon.dahyoung@gmail.com

ABSTRACT

Simulation and artificial intelligence (AI) are essential in developing, operating, and maintaining aerospace technology. This aerospace domain requires a large-scale, efficient, realistic, and optimized virtual environment. Among all components of the environment, buildings pose the most significant challenge due to their façade diversity, geometrical complexity, and spatial dominance. Although various approaches-such as building information modeling (BIM), geographic information system (GIS), and procedural methods of game industry-have been introduced to resolve these issues, but limitations persist in computational load, resource requirements, and quality for real-time aerospace simulation workflows.

This study proposes an automated framework for generating level of detail (LOD) 3 building meshes across kilometer-scale regions. The approach integrates BIM and GIS concepts by leveraging data from digital topographic maps, digital cadastral maps, the Building Act, and PBR material catalogs. With the information derived from the data, building meshes are created in a way game assets are built. Tools such as Python, Rhino Grasshopper, Blender, and Unreal Engine are sequentially employed to preprocess data, model and texture building components, and render the virtual environment.

Qualitative and quantitative evaluations are conducted with the buildings and the virtual surroundings. Simulated scenes from the virtual environment corresponding to real-world regions are provided as illustrative examples. Additionally, synthetic data generated from the virtual environment were trained in U-net model for the purpose of segmenting roads and buildings. The evaluation confirms the effectiveness of the method in creating 3D buildings to empower simulation and synthetic data workflows.

ABOUT THE AUTHORS

Goeun Yeo majors in Architecture and has a profound interest in automating architectural processes. While working at SIM2REAL for two years, she focuses on interpreting the real world into the virtual world. Having expertise in GIS engineering and 3D modeling, she aims to assist simulation and synthetic data technologies by integration of interests.

Mungi Kim holds a degree in Radio and Information Communication Engineering and has been working at SIM2REAL for two years, specializing in virtual environment development and simulation system implementation using Unreal Engine. Through expertise in simulation technology, he enhances the realism and functionality of virtual experiences.

Youngjun Sim holds a degree in Computer Engineering and has three years of experience at SIM2REAL, specializing in virtual environments. He focuses on procedural generation to efficiently create high-quality, large-scale virtual worlds. His work supports research in drone and vehicle simulation, as well as synthetic data generation for AI training.

Dahyoung Jeon holds a master’s degree in Aerospace Engineering with a focus on Modeling and Simulation. For the first 12 years of his professional career, he supported flight tests and validation through simulation at the Korean Air R&D Center. He founded SIM2REAL in 2022 in South Korea to solve data problems and simulation validation for artificial intelligence. Have patented and validated synthetic data and simulations in various fields, and is currently developing simulation environment optimization and automation, especially for aerospace.

Virtual Environment for Aerospace Simulation and AI Data: Focused on Automatic Building Generation

Goeun Yeo, Mungi Kim, Youngjun Sim, Dahyoung Jeon
SIM2REAL
Daejeon, South Korea
edisin67@gmail.com, kim.mungi@sim2real.co.kr, slkafsim@gmail.com, jeon.dahyoung@gmail.com

INTRODUCTION

Prospects and limitations of aerospace industry development using simulation and AI are attracting considerable attention. As shown in Figure1, SIM2REAL has collaborated with various aerospace companies on projects that exemplify these opportunities and challenges. For example, a project conducted by the Korea Aerospace Research Institute (KARI) simulated drone flight while checking correspondence between the real world and the virtual environment. Additionally, a project with CONTEC enhanced AI model detection for martial targets with the synthetic satellite data driven from virtual environments. Projects like these show the advantages in testing abnormal situations and harnessing additional data, but the drawbacks occur with development time and quality in creating a virtual environment, and computer performance issues related to hardware specifications.

Despite the challenges, virtual environments are unreplaceable or even suggest new opportunities. Although the generative process of data generation is coming forward, issues about data bias and model autophagy disorder (MAD) are unresolved. Environment with individual objects can implement a fully visual interactive simulation, which allows destruction of buildings, aircraft landing, and more. While buildings are the core ingredient of virtual space and interaction, they are hard to find a match of need from the ready-made assets, and their complexity and size call for more sources in computation. Many automated processes having the need of high-cost data or man-made fundamental assets, this study suggests an automatic building generation method with less input and verifies the method via simulation and synthetic data.

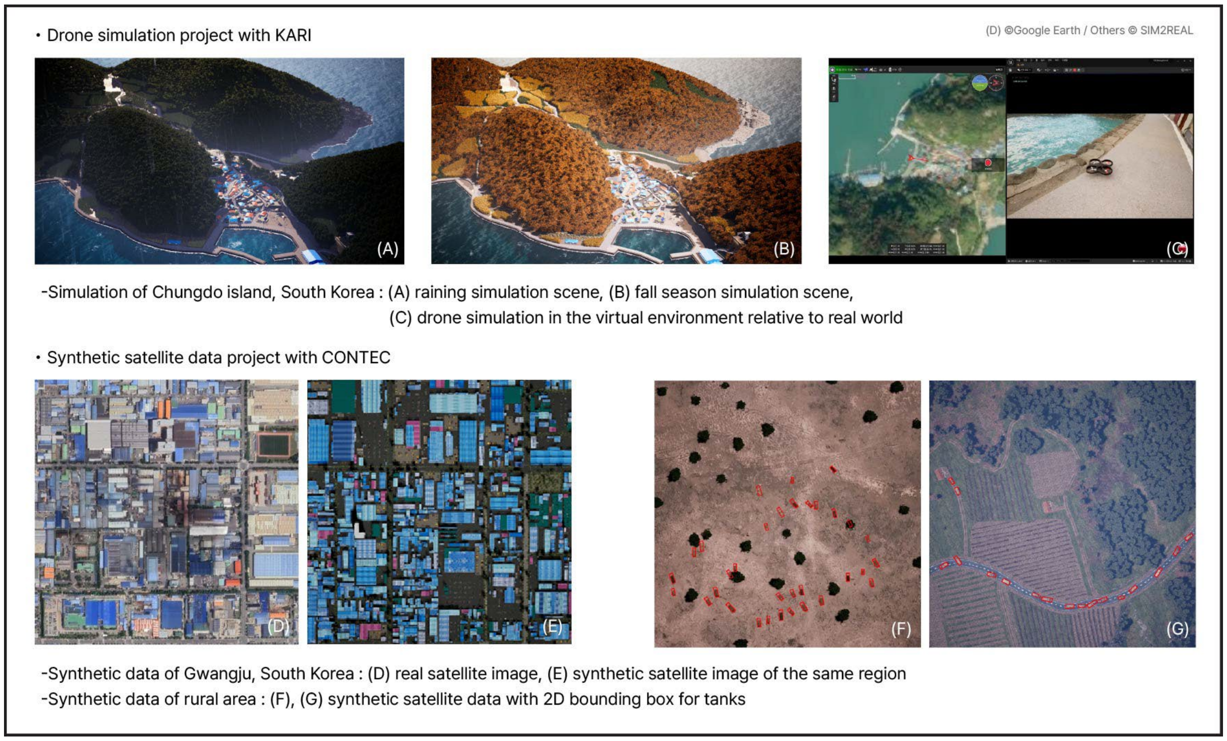

• Drone simulation project with KARI
(D) ©Google Earth / Others © SIM2REAL
(A)
(B)
(C)
-Simulation of Chungdo island, South Korea : (A) raining simulation scene, (B) fall season simulation scene,
(C) drone simulation in the virtual environment relative to real world
• Synthetic satellite data project with CONTEC
(D)
(E)
(F)
(G)
-Synthetic data of Gwangju, South Korea : (D) real satellite image, (E) synthetic satellite image of the same region
-Synthetic data of rural area : (F), (G) synthetic satellite data with 2D bounding box for tanks

Figure 1. Drone Simulation and Synthetic Satellite Data

PREVIOUS RESEARCH

Buildings for Aerospace Simulation and Synthetic Data Generation

Simulation and AI development for aerospace domain are implemented for various use cases, significantly advancing in recent years. Though simulation and synthetic data generation for AI model training have many differences, both share a same perspective in trying to foresee and prepare for possible outcomes, thus gaining safety and effectiveness in the development process and the results. Ground control simulation for air traffic management (ATM) and unmanned aerial vehicle (UAV) require complex environments that resemble the real world such as the digital twin frameworks for simulating autonomous drone operations at airports like London Heathrow (Conrad et al., 2023), (Zhang et al., 2024), (Turco et al., 2024). Flight simulation requires realistic references within large areas to test aircraft dynamics, control systems, and pilot training. Integrating multiple tools for UAV simulation enhances sensor fidelity and enables realistic movement of objects, supporting the development of autonomous UAV systems (Tejero-Ruiz et al., 2024). The Future Systems Simulator (FSS) at Cranfield University suggests a modular, user-centered simulator designed for research in human factors, single-pilot operations, and alternative control approaches (Korek et al., 2024). As for synthetic data generation, it is handling data scarcity and high costs of real data to train AI models and test complex systems. Examples show effectiveness, such as RCNN model trained purely on synthetic data that achieved high accuracy in real-world drone detection tasks (Wiśniewski et al., 2024), and even aims on creating a world model for UAV, with the use of synthetic data extracted from the simulation platform (Yao et al., 2024).

With so much usage and precedent results, it is evident that virtual environment for simulation and synthetic data is crucial for aerospace domain. Among figures that populate the virtual environment - terrain, roads, buildings, vegetation, street props, vehicles, and more - this study focuses on buildings that are the most work-loaded and computationally heavy objects. Though there are few research that depicts the characteristics of the buildings for simulation and synthetic data for aerospace domain, we can infer from the related ones. Traits like real time synthesis, cognitive model integration, database-driven design, visual fidelity, adaptability are suggested via simulation (РОГАНОВ et al., 2023). Randomization, level of detail, scalability are suggested for synthetic data generation (Fedorova et al., 2021). Thereby, the author suggests seven characteristics for the automatically generated buildings of this paper: time-saving, high-quality, optimized, individual, diverse, large-number, and minimal-dependence.

Different Automatic Building Generation Concepts and Methods

Among numerous automatic building generation methods, BIM, GIS, and game perspective offer relevant solutions. BIM serves a key role in Architecture, Engineering, and Construction (AEC) industry, with its perspective not only defines the building shape and composition but also intervenes in the whole lifecycle of the building. From design and construction to maintenance and demolition, BIM aims to represent everything of the building. With much data to be organized, Industry Foundation Classes (IFC) standard from the buildingSMART (buildingSMART, 2024) comes in hand, which is the most widely used open data schema that defines a geometry in boundary-representation (b-rep), constructive solid geometry (CSG), and sweep volumes (Zhu et al., 2018). Creation of BIM model for the public office building under construction via Autodesk Revit well shows the usage of IFC standard, with multiple classes and definitions integrated to digital twin the building (Dai et al., 2024). Buildings of GIS focus not on individual, but the collection of buildings within large areas. From the characteristic that GIS interests in every attribute of the Earth-landscapes, man-made objects, and even chemical properties, properties of already-built buildings are collected as data within geospatial context. City Geography Markup Language (CityGML) standard of the Open Geospatial Consortium (OGC) (OGC, 2022) is an open standard data model and exchange format to store 3D models of cities and landscapes, and geometries are only defined in b-rep (Zhu et al., 2018). Examples of CityGML buildings are derived from vector data or point cloud data or even converted from BIM models (Ohori et al., 2018).

Integration of BIM and GIS to generate buildings has been discussed profoundly, with each domain having advantages in detail and spatial extent. From old to new research on the integration of the two, BIM information has been converted to match the GIS data format, but data conversion problems have been continuously provoked in application and data sectors. Differences in class numbers, dissimilar class arrangements, geometry creation methods, and georeferencing system hinder further progress even though IFC standard and CityGML standard are both considering each other on their updates (Isikdag et al., 2009), (Zhu et al., 2018), (Tan et al, 2023), (Ahmad et al., 2024). Though many drawbacks, full conversion and integration of IFC to CityGML has been completed through feature manipulation engine (FME), resulting a virtual environment visualized with Unreal Engine5 and Cesium Ion (Lam et al., 2024).

Buildings for game assets are automatically generated via concept named procedural generation of buildings (PGB). PGB to create common buildings in human settlements can be classified with human intervention, generative capabilities, and core technologies. Game assets partially have a nature of artistic outputs, so many methods are not fully automatic but open to opportunities of modification with the repeat of generating and testing. As for the generative capabilities, there are few that support the generation of full buildings from interior to exterior, and even if it does support the generation of full buildings or all the exterior, available forms of buildings are limited. Many technologies are implemented, from shape-based grammar to machine learning, but machine learning techniques yet abide in 2D forms. Methods with elementary objects and predefined building blocks assembled with unique grammar, label-based selections, or complex tree operations are currently dominant. Generated buildings with the stated methods can be divided as six components - outer walls, roofs, texturing, floor plans, furnishings, and stairs - which PGB aims to make game-compatible, optimized, and deformable (Kutzias et al., 2024).

Suggested Automatic Building Generation Method of the Paper

Though many automatic and procedural methods have been developed and optimized for 3D building generation as above, none of them match all seven characteristics of the buildings that have been previously suggested. This project focuses on generating 3D buildings with minimal data by integrating BIM, GIS, and game perspective of automatic building generation to solve the issue. BIM can handle characteristics of high-quality and minimal-dependence with the usage of large language model (LLM) defining the principles of building structure and composition. Examples of rule-based modeling can be found as planning regulations and building permit regulations to create and check BIM models (Koutamanis, 2024), (Hobeika et al., 2022). GIS handles data for buildings within km-scale regions which offers large-number and time-saving characteristics. Vector data accessible from multiple sources offer reference to the real world. Game perspective can structure the parts of the building to create optimized, individual, and diverse building meshes that can be directly put into the virtual environment operated with game engines.

To convey all three perspectives in one workflow, the developing process of this paper includes analyzing data, creating mesh, texturing mesh, assembling virtual environment, and generating synthetic data. Python, with its extensive libraries for geographical data processing and analysis, serves as the primary tool for data analysis in the workflow. Rhino8 Grasshopper is utilized for building mesh modeling, making use of numerous add-ons and procedural tools along with Python support through GHpython component. Blender4.2 facilitates efficient UV mapping and materials through open source platform and Blender Python API (Blender, 2025) which handles textures and exports the final 3D buildings in the process. Unreal Engine5 renders the virtual environment with the generated buildings, enabling both simulation and synthetic data generation.

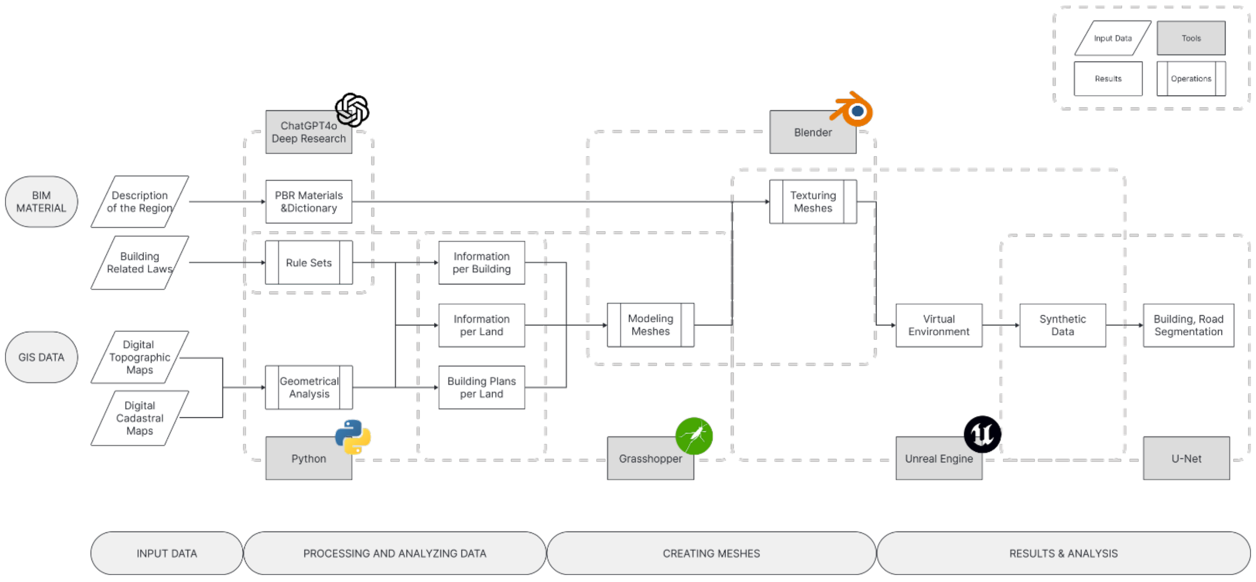


Figure 2. Suggested Full Process for Automatic Building Generation

PROCESSING AND ANALYZING DATA

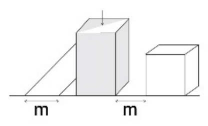
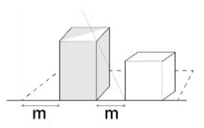
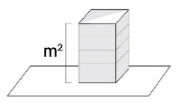
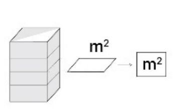
This chapter aims to translate data into information used for creating 3D building meshes. Apart from the traditional method of integrating BIM and GIS as data conversion, the author conveys the integration in a rule-based way that not only demands less data sources but also expands applicability for diverse regions. For the adaptation of BIM to the process, the Building Act of South Korea, the Enforcement Decree of the Building Act of South Korea, and the Building Ordinance of the specific region were evaluated with the LLM, ChatGPT4o, creating rule-sets. Additionally, royalty-free PBR materials were also collected via the LLM from the Web with the description of the region's building type, which are later used as material resources for the buildings. For GIS data, digital topographic maps from the National Geographic Information Institute (NGII) (https://www.ngii.go.kr/kor/main.do), and digital cadastral maps from the V-World (https://www.vworld.kr) were used to provide real world data for the geospatial analyzation. Rule-sets of BIM and geospatial analyzation of GIS are then combined to generate mesh-generation-ready information. In the overall process of data processing, Python version 3.13.1 with the internal library re, random, math, collections, and the external library geopandas 1.0.1, shapely 2.0.6, pandas 2.2.3, and typing 4.12.2 was used.

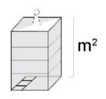
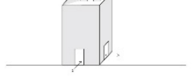
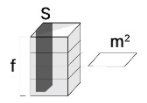
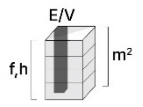
Lingual Data for BIM

Previous research suggests that LLMs have an ability to translate raw data into expert rules (Noever et al., 2023), and have basic understanding of BIM (Choi et al., 2023). The prompt put into ChatGPT4o deep research included Building Act of South Korea, Enforcement Decree of the Building Act of South Korea, and Building Ordinance of Milyang, South Korea and Gwangju, South Korea achieved from the Korea Law Information Center (https://www.law.go.kr/). Like in Table1, the rule-sets, which are python functions, determine building composition such as building height, setback distance, max floor area ratio, window area, estimated number of occupants, properties of doors, properties of stairs, and properties of elevators.

PBR materials were collected via the LLM to texture building meshes. Categories for the materials were first decided, each corresponding to the building component, which are flat roof, non-flat roof, wall, column, slab, door, glass, door frame, and window frame. Descriptions of the region's buildings were also given as material style guidelines. From the categories and the descriptions, the LLM provided links of free PBR materials for the region, a dictionary with all the names for all searched PBR material, and a dictionary with each material's estimated real-world size.

Table1. Building Related Laws Translated Rule-sets

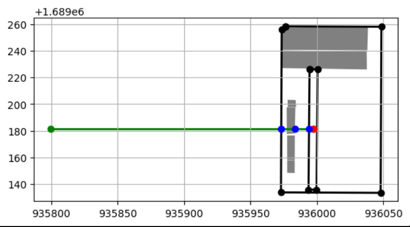
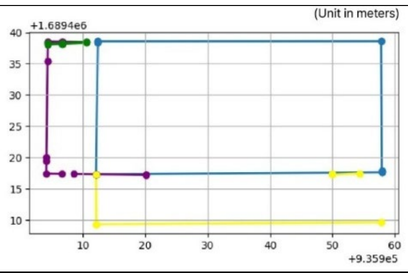
Rule-set Name	Objective	Input Parameters (data type)	Output Results (data type)	Principle Diagram
calc_building_height	Return maximum building height within specific land	zone (string), road_width (float), north_setback_dist (float)	max_height (float)	
calc_setback_distance	Return minimum distance from other building or land boundary	adjacent_type (string), height (float), other_height(float), facing_windows(bool)	setback_dist (float)	
get_max_fsi	Return maximum gross area ratio for specific land	zone (string)	max_ratio (float)	
calc_window_area	Return minimum window area per floor for lighting and ventilation	floor_area (float), for_lighting (bool), for_ventilation (bool)	window_area (dict)	

estimate_occupants	Return estimated capacity for a building	building_use (string), floor_area (float), num_units(integer)	occupants (integer)	
calc_exit_doors	Return number of doors and the total width of doors.	num_people (integer)	door_prop (tuple)	
calc_stairs	Return number of stairs and the minimum width for each stair.	num_floors (integer), max_floor_area (float), building_use (string)	stairs_prop (tuple)	
calc_elevators	Return number of elevators and the types of each elevator.	num_floors (integer), total_area (float), building_height (float), building_use (string)	elevators_prop (dict)	

Vector Data from GIS

GIS data offers input parameters for the rule-sets of BIM. While some parameters are specified in the vector data itself and only need light conversion, some need to be estimated through geometrical methods. Building polygons of digital topographic maps and land polygons of digital cadastral maps are geometrically analyzed as suggested in Table2 to estimate building forms and its surroundings as numerical values.

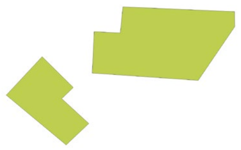
Table 2. Examples of Geometrical Analysis

Geometrical method	Input Data	Output Result	Analysis Example
Intersection	building center point, land width, building polygons, land polygons, building bounding box	distance to other buildings, properties of adjacent buildings, distance to land boundary, properties of adjacent lands	
Buffer	building ID, buffer distance for doors, buffer distance for windows, building polygons	applicable range of windows and doors in length of polygon segments	

Information for Creating Meshes

Data derived from GIS data were put into the rule-sets and the results of the rule-sets were post-processed to match the data format for the modeling process. Through the overall process, GIS data served as a base data for a specific km-scale large region, enabling resemblance with the real world that contains over thousands of buildings. Rule-sets from the BIM perspective generated specific data than what GIS data can offer, enabling more detailed, high-quality buildings without additional scanned or measured data. As in Table3, the entire result of data processing consists of one shapefile (.shp) that indicates a base shape of the building plan, one text file (.csv) that explains broad characteristics for each building on a land basis, and one text file (.csv) that explains specific properties for each building.

Table 3. An Example of Data Processing Output

File	Shapefile	Text File per land			Text File per building		
Parameter type	building shapes for each land	index	type	composition	index	numeric	domain
Paramters		Index, LandNum, BuildingID	Ltype, Btype, Rtype	Floors, Height, Setback	Index, BuildingID, Floor	Height, Zpos, Area, Setback	Door, Window, WinArea

CREATING MESHES

While some conventional methods create the whole scene as one object, for further physical interaction with the virtual environment and to-be-added actors, building generation method for creating game assets are adapted for individual, diverse meshes. All buildings were modeled on a component basis, each building containing walls, columns (if piloti was assigned), slabs (for each floor), doors (with frames and doors), and windows (with frames and glass). Leveraging previously processed building information, this method supports the generation of multiple variants for each building. To enhance simulation performance, all building models were optimized by reducing vertices in the modeling process.

Modeling

Building 3D meshes was conducted via Rhino Grasshopper as b-rep, based on the information processed and exported previously. With the suggested process, 3D buildings that follow the exact position and building plan were created, with an opportunity of randomness filling the missing pieces between what the data has defined. With three 0~5000 ranged main variables, Grasshopper embedded genetic algorithm was implemented to create the best shaped building within opportunities. LunchBox and Pancake add-on were used for mesh optimization and mesh exportation in FBX format. The automatic process of modeling meshes is the same as the following explanation of order.

① Import Information: import text files (.csv) and vector files (.shp).

② Parse Information: match geometries from vector files and properties from text files in a format compatible for the Rhino Grasshopper modeling workflow.

③ Model Roofs: model the roof that has been assigned to the building, using roof height and building plan. Roofs can be flat, gable, half-hip, full-hip, or gable-hip.

④ Model Walls: model the wall of the building using floor height, building height, and building plan.

⑤ Model Pilotis: for piloti assigned buildings, create pillars instead of walls for the first floor using floor height and building plan.

⑥ Model Openings: create masses that indicates position and size for doors and windows, with the information about possible door domain, possible window domain, total door number and width, and window area per floor.

⑦ Model Doors: two types of doors are modeled, one-sided and two-sided, to match the door openings. Each door consists of a door frame and door(s).

⑧ Model Windows: three types of windows are modeled-picture window, sliding window, and casement window. Each window has window frames and glass.

⑨ Optimize Meshes: every created meshes are decimated, reducing vertices to nearly half of the original.

⑩ Export FBX: all modeled parts are divided into layers that indicates materials, and then exported as FBX file.

Texturing

Texturing process was carried out via Blender Text Editor with Blender Python API. All building meshes from the previous step were processed automatically as stated below.

① Prepare PBR Materials for Each Category: with links that have been collected in the PROCESSING AND ANALYZING DATA, PBR materials are automatically created and saved in Blender Asset Library.

② Import FBX files: import text files (.csv) and vector files (.shp).

③ Assign Materials: Materials for each mesh are randomly selected in a pool of materials for each category.

④ UV mapping: analyze mesh surface UV and the texture's x, y size and rotate the UVs so that the long sides correspond to each other. For scale, adjustments are made with previously estimated materials' real-world size compared with the meshes' real-world size.

⑤ Export GLB: join meshes of each building and export each building in light and unreal compatible 3D format (.glb).

Final Generated Buildings

As attached in Table4, generated buildings from the rural suburban region of Milyang, South Korea and the industrial region of Gwangju, South Korea are in LOD3, according to the OGC CityGML 3.0 (OGC, 2021), attatched as Figure3.

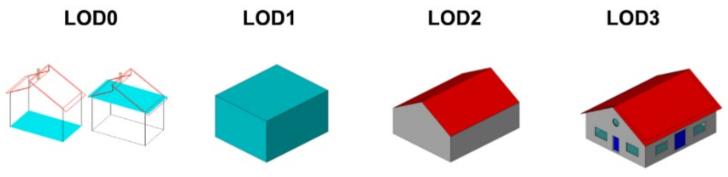
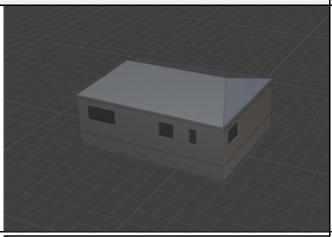
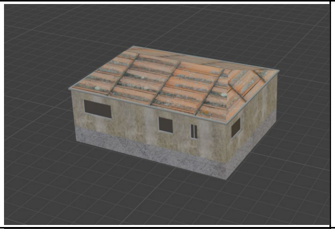
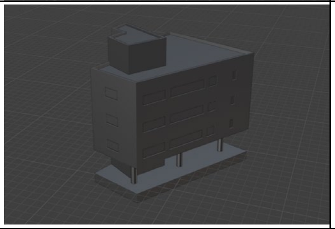
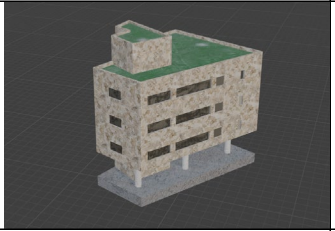
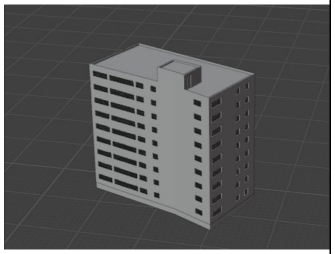
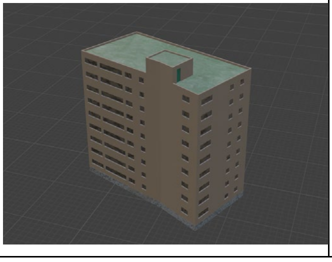
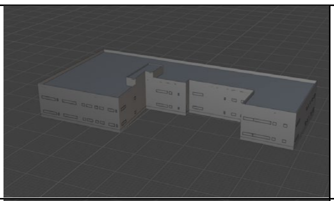
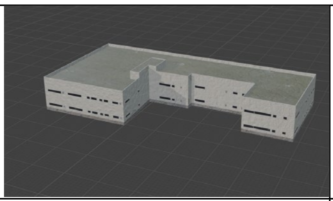
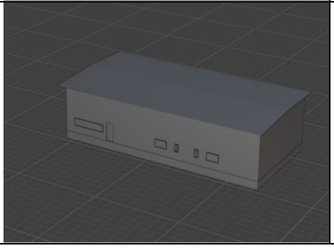
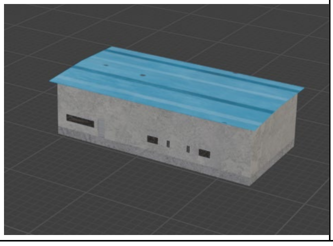


Figure 3. Representation of OGC CityGML3.0 LOD Levels

Table 4. Generated Buildings by Type

Building Type	Region	Modeled Mesh	Textured Mesh	Characteristics
Detached House	Milyang			-half-gable roof -one story house -low ceiling height -one-sided door -textures for each building component
Multi-family House	Milyang			-flat roof -piloti for the first floor -staircase room on the roof -two one-sided doors -textures for each building component
Apartment Housing	Milyang			-flat roof -staircase room on the roof -nine stories building -two-sided door and one-sided door -textures for each building component

Factory	Gwangju			-flat roof -two stories building -two one-sided doors -textures for each building component
Warehouse Facility	Gwangju			-gable roof -one story building -high ceiling height -one-sided door -textures for each building component

*All images were captured in Blender viewport.

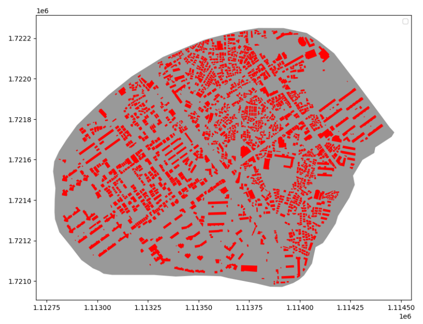
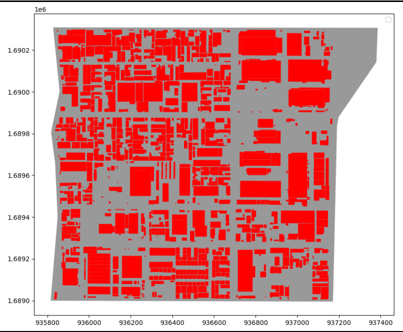
RESULTS

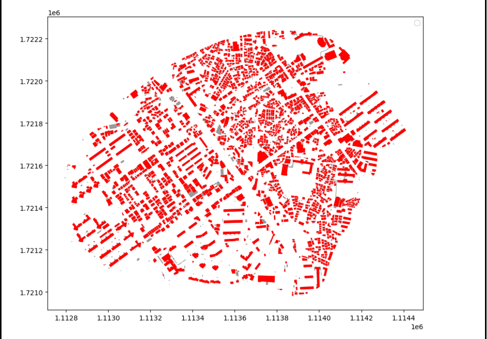
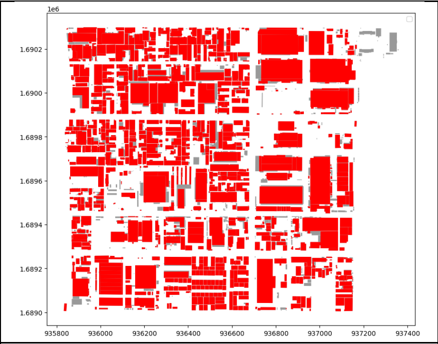
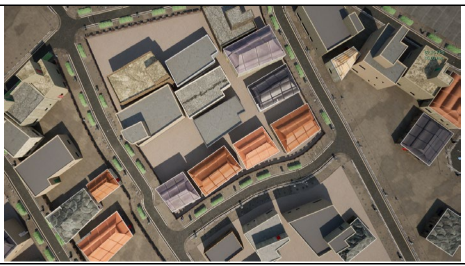
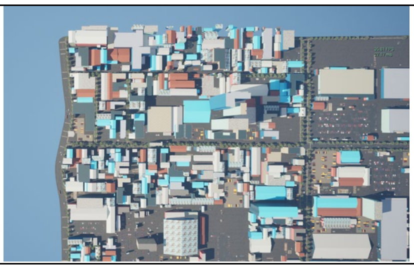
Two virtual environments rendered in Unreal Engine5 with the generated building from the process were constructed. Both environments have a real-world counterpart, regions from Milyang, South Korea and Gwangju, South Korea. Milyang region is a rural suburban area where common living facilities are dominant. Gwangju region is a typical Korean industrial region, with many factories and warehouses packed in the outskirts of the city. Synthetic data was generated within the virtual environment derived from the region of Gwangju, South Korea.

Virtual Environment

Virtual environment is composed of buildings, roads, plots, and props. For this paper, buildings were automatically generated with the previously explained method, and the other components were manually imported, generated and placed. As mentioned in Table 5, building generation was held over large quantities but managed to obtain a frame rate available for real time operation even in an aerial point of view. From all the buildings of the vector data, buildings within the target area were first selected, and buildings below area 15m^2 and buildings without walls were excluded. Some buildings were additionally dropped in the process due to insufficient information and mesh problems.

Table 5. Overall Properties of Each Region

Step	Properties	Milyang Region	Gwangju Region
PROCESSING AND ANALYZING DATA	Size of the Area (km^2)	1.544	1.823
	Target Area and Buildings within Area	 1e6 1.7222 1.7220 1.7218 1.7216 1.7214 1.7212 1.7210 1.11275 1.11300 1.11325 1.11350 1.11375 1.11400 1.11425 1.11450 1e6	 1e6 1.6902 1.6900 1.6898 1.6896 1.6894 1.6892 1.6890 935800 936000 936200 936400 936600 936800 937000 937200 937400
	Total buildings within Area	3746	2070
	Executed Buildings	2318	1073

	All Buildings within Area and Executed Buildings		
	Dominant Building Type	single-family house	factory
RESULTS	Simulation Scene in Aerial Perspective with Frame Rate		
	Characteristics of the Region	urban area with housings and downtowns	industrial area
	Unreal Engine Frame Rate	available for real time simulation (over 30 fps)	available for real time simulation (over 30 fps)

Synthetic Data

With the virtual environments above, aerial synthetic data was automatically extracted by prepossessed technology of SIM2REAL. As in Table 6, the data has three classes in total and the image format is PNG (.png). As shown in Figure4, the RGB image and the segmentation image for roads and buildings were extracted.

Table 6. Synthetic Data Properties

Properties	Data
Quantity	3600 images
Size	512X512 pixels
Resolution	25cm
Annotation	Segmentation of buildings, roads, and others.

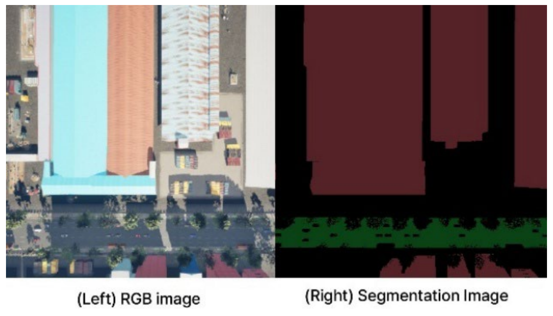


Figure 4. Data Example

ANALYSIS

For quantitative analysis, comparison of model train results with and without synthetic data was conducted. The AI model training was executed based on U-net architecture and real aerial data from the Korean AI platform, AIHub (AIHub, 2022). Three cases were trained on cuda platform as shown in Table 7, case1 being the identical setting with the AIHub to check the existing result in the local environment. Case2 was conducted to match the number of classes of the synthetic data, which merged all other classes except road and buildings, showing improvements in accuracy due to the inclusiveness of class3 (none classified area). Case3 was trained with additional synthetic data which was 10% of the real data. When case2 and case3 compared, accuracy in class1(building) improved, and remained similar in class2 and class3, resulting in a slight improvement in overall pixel accuracy. IOU for each class and mIOU also showed improvements when trained with synthetic data.

Table 7. U-Net Train Result

Category	Conditions	Case1. Default Setting	Case2. Merge Annotation	Case3. Add Synthetic Data
Train Data	Real Data	36000	36000	36000
	Synthetic Data	None	None	3600
Validation data	Real Data	4000	4000	4000
Test Data	Real Data	5000	2000	2000
Train Parameter	Min Epoch	100	100	100
	Max Epoch	200	200	200
	Batch Size	8	8	8
	Learning Rate	0.001	0.001	0.001
	Minimum Accuracy	0.95	0.95	0.95
	Annotation	11 classes (Building, Parking Lot, Road, Street Tree, Paddy Field, Greenhouse, Dry Field, Deciduous Forest, Coniferous Forest, Bare Land, Water Body, None Classified Area)	3 classes (Building, Road, None Classified Area)	3 classes (Building, Road, None Classified Area)
Train Result	Result epoch	200	100	100
	Overall Pixel Accuracy	**0.9211**	**0.9904**	**0.9905**

	Accuracy of Each Class	**class 1 = 0.9315**
		class 2 = 0.8412
		class 3 = 0.9426
		class 4 = 0.6532
		class 5 = 0.9675
		class 6 = 0.9547
		class 7 = 0.8761
		class 8 = 0.8653
		class 9 = 0.6812
		class 10 = 0.8527
		class 11 = 0.917
			class 1 = 0.9354
			class 2 = 0.9526
			class 3 = 0.9954
				class 1 = 0.9425
				class 2 = 0.9528
				class 3 = 0.9950

	IOU of each class	**class 1 = 0.8778**
		class 2 = 0.7593
		class 3 = 0.9036
		class 4 = 0.5161
		class 5 = 0.9333
		class 6 = 0.9118
		class 7 = 0.8012
		class 8 = 0.7309
		class 9 = 0.5591
		class 10 = 0.7449
		class 11 = 0.8615
			class 1 = 0.8910
			class 2 = 0.9162
			class 3 = 0.9895
				class 1 = 0.8922
				class 2 = 0.9175
				class 3 = 0.9896

	mIOU	**0.7817**	**0.9322**	**0.9331**

FINDINGS AND CONCLUSIONS

The primary objective of this study was to develop an automated framework for generating 3D building meshes that satisfy essential requirements for simulation and synthetic data applications - time-saving, high-quality, optimized, individual, diverse, large-number, and minimal-dependence on external datasets. This automated process places particular emphasis on the data processing part, where the author suggested a relatively new perspective of integrating BIM and GIS concepts via game asset creation workflow. The integration of BIM rule-sets enabled precise and detailed modeling of buildings without the need for additional data sources, while GIS data facilitated the large-scale creation of thousands of building models with real-world spatial context. Adopting a game-asset-oriented perspective through the use of Rhino Grasshopper and Blender allowed procedural generation of building assets with a wide variety of shapes, optimized for real-time simulation environments.

Individual building assets implemented in game engines exhibit physical simulation capabilities, enabling interaction between objects such as building demolition or drone landing. While advanced post-processing and sophisticated data augmentation techniques were not applied due to time constraints, the training results demonstrate that the proposed automated generation process can effectively produce basic synthetic data and facilitate the creation of edge-case scenarios. Furthermore, the building models possess additional potential for improvement through the evaluation of more comprehensive building regulations, enhanced procedural modeling of components, and deeper manipulation of mesh UVs, which will be explored in future work.

REFERENCES

Conrad, C., Fremond, R., Mukherjee, A., Delezenne, Q., & Tsourdos, A. (2023). *Co-simulation Digital Twin Framework for Testing Future Advanced Air Mobility Concepts: A Study with BlueSky and AirSim*. https://doi.org/10.1109/dasc58513.2023.10311124

Zhang, S., Wang, H., Zhai, M., Yi, P., & Meng, C. (2024). Research on experimental evaluation model of unmanned aerial vehicle autonomy based on digital twin. *Journal of Physics*, *2791*(1), 012075. https://doi.org/10.1088/1742-6596/2791/1/012075

Turco, L., Zhao, J., Xu, Y., & Tsourdos, A. (2024). *A Study on Co-simulation Digital Twin with MATLAB and AirSim for Future Advanced Air Mobility*. 1–18. https://doi.org/10.1109/aero58975.2024.10521333

Tejero-Ruiz, D., Pérez-Gran, F. J., Viguria, A., & Ollero, A. (2024). *Realistic Unmanned Aerial Vehicle Simulation: A Comprehensive Approach*. 1–6. https://doi.org/10.1109/robot61475.2024.10796915

Korek, W. T., Beecroft, P., Lone, M., Bragado Aldana, E., Mendez, A., Enconniere, J., Asad, H. U., Grzedzinski, K., Milidere, M., Whidborne, J., Li, W.-C., Lu, L., Alam, M., Asmayawati, S., Del Barrio Conde, L., Hargreaves, D., & Jenkins, D. (2024). Simulation framework and development of the Future Systems Simulator. *Aeronautical Journal*. https://doi.org/10.1017/aer.2024.91

Wiśniewski, M., Rana, Z. A., Petrunin, I., Holt, A., & Harman, S. (2024). *Drone Detection using Deep Neural Networks Trained on Pure Synthetic Data*. https://doi.org/10.48550/arxiv.2411.09077

Yao, F., Yue, Y., Liu, Y., Sun, X., & Fu, K. (2024). *AeroVerse: UAV-Agent Benchmark Suite for Simulating, Pre-training, Finetuning, and Evaluating Aerospace Embodied World Models*. https://doi.org/10.48550/arxiv.2408.15511

РОГАНОВ, Р., МИХЕЕВ, Ю., ЧЕТВЕРГОВА, В., & Sun, B. (2023). *Features of synthesis of visually observable during flight 3d-models of the external environment for a flight simulator. 10*. https://doi.org/10.25791/pribor.10.2023.1446

Fedorova, S., Tono, A., Nigam, M. S., Zhang, J., Ahmadnia, A., Bolognesi, C. M., & Michels, D. L. (2021). Synthetic 3D Data Generation Pipeline for Geometric Deep Learning in Architecture. *arXiv: Computer Vision and Pattern Recognition*. https://doi.org/10.5194/ISPRS-ARCHIVES-XLIII-B2-2021-337-2021

BuildingSMART (2024). IFC 4.3.2.0 (IFC4X3_ADD2), https://standards.buildingsmart.org/IFC/RELEASE/IFC4_3/

Zhu, J., Wright, G.L., Wang, J., & Wang, X. (2018). A Critical Review of the Integration of Geographic Information System and Building Information Modelling at the Data Level. *ISPRS Int. J. Geo Inf., 7*, 66.

Dai, C., Cheng, K., Liang, B., Zhang, X., Liu, Q., & Kuang, Z (2024), Digital twin modeling method based on IFC standards for building construction processes, Frontiers in Energy Research Volume 12 – 2024, 10.3389/fenrg.2024.1334192

Open Geospatial Consortium (OGC) (2021). OGC City Geography Markup Language (CityGML) 3.0 Conceptual Model Users Guide, https://docs.ogc.org/guides/20-066.html#overview-section-levelsofdetail

Ohori, K., Biljecki, F., Kumar, K., Ledoux, H., & Stoter, J. (2018). *Modeling Cities and Landscapes in 3D with CityGML* (pp. 199–215). Springer. https://doi.org/10.1007/978-3-319-92862-3_11

Isikdag, U., & Zlatanova, S. (2009). Towards Defining a Framework for Automatic Generation of Buildings in CityGML Using Building Information Models. 10.1007/978-3-540-87395-2_6.

Zhu, J., Wright, G., Wang, J., & Wang, X. (2018). A Critical Review of the Integration of Geographic Information System and Building Information Modelling at the Data Level. *ISPRS International Journal of Geo-Information*, *7*(2), 66. https://doi.org/10.3390/ijgi7020066

Tan, Y., Liang, Y., & Zhu, J. (2023). CityGML in the Integration of BIM and the GIS: Challenges and Opportunities. *Buildings*, *13*(7), 1758. https://doi.org/10.3390/buildings13071758

Ahmad, A., Abdul Maulud, K. N., & Syed Abdul Rahman, S. A. F. (2024). Method and Use in Implementation of Integration BIM and GIS and Its Potential in the Drainage System Management: A Review. *Jurnal Kejuruteraan*. https://doi.org/10.17576/jkukm-2024-36(2)-10

Lam, P.-D., Gu, B.-H., Lam, H.-K., Ok, S.-Y., & Lee, S.-H. (2024). Digital Twin Smart City: Integrating IFC and CityGML with Semantic Graph for Advanced 3D City Model Visualization. *Sensors*, *24*(12), 3761. https://doi.org/10.3390/s24123761

D. Kutzias and S. von Mammen, "Recent Advances in Procedural Generation of Buildings: From Diversity to Integration," in IEEE Transactions on Games, vol. 16, no. 1, pp. 16-35, March 2024, doi: 10.1109/TG.2023.3262507.

Koutamanis, A. (2024). Planning Regulations and Modelled Constraints in BIM: A Dutch Case Study. *Buildings*, *14*(4), 939. https://doi.org/10.3390/buildings14040939

Hobeika, N., van Liempt, J., Noardo, F., Arroyo Ohori, K., & Stoter, J. (2022). GeoBIM Information to Check Digital Building Permit Regulations, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLIII-B4-2022, 529–535, https://doi.org/10.5194/isprs-archives-XLIII-B4-2022-529-2022

Blender (2025). Blender 4.4 Python API Documentation, https://docs.blender.org/api/current/

Choi, J., Koo, B., Yu, Y., Jeong, Y., Ham, N. (2023). Evaluating ChatGPT's Competency in BIM Related Knowledge via the Korean BIM Expertise Exam, *Journal of KIBIM*, Vol.13, No.3, 21-29.
(Choi, Koo, Yu, Jeong, & Ham, 2023)

Noever, D., Regian, W. (2023). How Large Language Models Translate Raw Data into Expert Rules, 2023 Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC), No.23139

AIHub (2022). Aerial and Satellite Images for Land Cover Mapping,
https://www.aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=data&dataSetSn=71361