SimBuild
2024

Eleventh National Conference of IBPSA-USA
Denver, Colorado
May 21 – 23, 2024

Rapid Building Feature Extraction and Geometry Formulation Using Machine Learning

SoumyaDeep Chowdhury, Kuljeet Singh Grewal
Future Urban and Energy Lab for Sustainability (FUEL-S), Faculty of Sustainable Design Engineering (FSDE), University of Prince Edward Island,
550 University Ave, Charlottetown, PE, Canada C1A 4P3

Abstract

Increasing energy consumption has led to wide-scale optimization efforts of urban and sub-urban environments. To supplement this, building-energy-modeling (BEM) methods are applied to provide insights and identify optimizations. Several inputs are required for BEM, including climate, usage, and most importantly geometric data. Creation of geometric data is a time-consuming endeavor. Methods of automation often incorporate expensive or not widely available technology such as light detection and ranging (LiDAR) or unmanned aerial vehicle (UAV) imagery. This document aims to provide and prove the viability of a methodology to create building models at high levels-of-detail (LOD), using only readily available sources such as OpenStreetMaps (OSM) and street-view images (SVIs). Modern image processing techniques and machine learning algorithms such as convolutional-neural-networks (CNNs) and regional-convolutional-neural networks (R-CNNs) are explored with the goal of creating building geometry model for energy modeling with machine learning algorithms reaching a precision of 71%, with geometry creation reaching an average accuracy of 95% compared to models made from on-site manual measurements. The process can easily be extended to be user-assisted to increase overall accuracy and reduce the time complexity of the workflow for building geometry creation outputting point-cloud data which will be ultimately applied for BEM.

Introduction

The creation of geometry for building energy-modeling existing buildings is a time-consuming task that requires a significant amount of tedious work. Methods of automation for this portion of the building energy modeling (BEM) process have been pursued, mostly revolving around the use of unmanned aerial vehicles (UAVs) (Maset et al., 2017), light detection and ranging (LiDAR) (O'Donnel et al., 2019), and photogrammetry (Gil et al., 2013). These data types are not easily accessible in certain areas, and can be expensive (Maset et al., 2017). Likewise, the identification of facade features can be an equally difficult endeavour. The use of streetview-images (SVI) in lieu of expensive unmanned aerial vehicle (UAV) flights has been pursued, but more complex facade information and precise roof forms has been neglected (Pang et al., 2022). There is a lack of methods for an openly available and automatic solution that provides a level of detail (LOD) 2 and beyond geometry including building height, roof shape, and facade features (windows, doors, garages) while using easily accessible and cost-effective data from street view images (SVIs). This work seeks to provide a methodology for the rapidly automated or semi-automated modeling of buildings at high levels of detail (LODs), using minimal data sources by applying modern image processing and machine learning techniques.

Background

Recent studies in the development of machine learning (ML) in computer vision have yielded a plethora of resources for such tasks, summarized by Gu. et al., 2017. Likewise, architecture for object detection using regional-convolutional-neural-networks (R-CNNs), and classification using convolutional-neural-networks (CNNs) have been improved with such architectures as *VGG16*, *ResNet-18*, and training methods such as transfer learning and data augmentation (Gu et al., 2017).

Other work has pursued the use of semantic segmentation to define skylines, using known values of the position of the camera in correlation with the building to retrieve building heights (Al-Habashna, 2022). Likewise, semantic segmentation (Dai et al., 2021) as well as object detection algorithms such as R-CNN, Faster R-CNN, Yolov4, Yolov5 (Sezen et al., 2022), have all been applied to detect facade features on buildings. Novel edge detection techniques have been developed specifically for the urban environment for aid in facade

detection and building identification (Orhei et al., 2021). These techniques have been researched and applied in isolation; however, a comprehensive process to apply photogrammetry, footprint data, semantic segmentation, object identification, classification ML, and geometry mapping has not been pursued in tandem to create high LOD building information models (BIM), including facade identification. The generated BIM-based geometry is intended for building energy modeling (BEM). The developed method can be used for the generation of building models in different LODs.

The output of this research is to outline a viable process to automate the geometry creation process, particularly focusing on the application of modern ML techniques to retrieve complex roof and facade features for building information modeling (BIM). This is accomplished by applying building segmentation methods (Dai et al., 2021), and applying the various advancements in machine vision CNN and R-CNN networks (Gu et al., 2018) to detect facade features and classify roof types to extract building proportions and facade information and create buildings with high LODs using minimal footprint and SVI information. The value of this work is to provide a rapidly automated or semi-automated (depending on desired accuracy) approach that will save time and effort in the development of geometry for BIM for BEM, allowing more focus to be placed on the interpretation and applications of the complete energy models. Furthermore, the intention is to accomplish this with the aforementioned data sources that are easily available, thereby reducing any financial barrier of entry to such efficiency. Ultimately, this work shows the application of modern machine learning and image processing techniques to reduce the work to create BIM for the purpose of BEM.

Methods

This section outlines the general overview of the proposed process, capturing the development process of the machine-learning (ML) algorithms and their application to provide BIM with LOD1 - LOD3 (Jaeger et al., 2018). The goal is to apply a combination of custom and pre-trained (for semantic segmentation) networks and develop discrete algorithms to retrieve additional information from the footprint and SVI to create a point-cloud to be applied to programs such as *Rhino-Grasshopper* to create a BEM. This work focuses on the step directly before BEM, to create and generate the time-consuming geometric data, and attempts to accomplish this in the form of a point-cloud. In total, two custom networks, with custom architecture are developed using MATLAB to detect objects (R-CNN) for facade features, and classify images (CNN) for roof classification are created, in addition to the discrete programming and algorithms to combine these networks to ultimately output point-cloud information. Another CNN is applied for more advanced roof classification; however, the hatching of advanced roof structures onto the developed point-cloud is not fully pursued. This will be addressed in further work. Although R-CNN and segmentation have been used in isolation to segment and detect facades in buildings, these processes have not been used in a holistic approach to creating building models themselves.

The process is applied to an OSM screenshot from the web browser; this step is taken to allow for interactivity in choosing houses for testing. The perspective side length from the road must ultimately be computed, while also converting the building footprints to point-cloud data. The perspective side length of a building is defined as the length of a building object across the plane parallel to the road, as shown in Figure 1. Thereafter, street

Figure 1: Road perspective side length

view images (SVIs) are utilized to retrieve the remaining data, such as the height of the building, roof shape, and facade features. Image augmentation, segmentation, and processing techniques are applied along with image convolution for the purpose of edge-detection. Using these methods, lines are categorized within the image to define the boundaries of the rightmost, leftmost, top, and bottom points of the building, along with the starting point of the roof. This process is reflected in Figure 2. To retrieve more complex information from the SVIs a number of ML algorithms must be applied. Firstly, to deal with the heavy vegetative and general noise present in sub-urban environments due to obstructions, semantic segmentation is implemented on the original image. This enhances the information extraction from the previous steps by removing lines outside of the

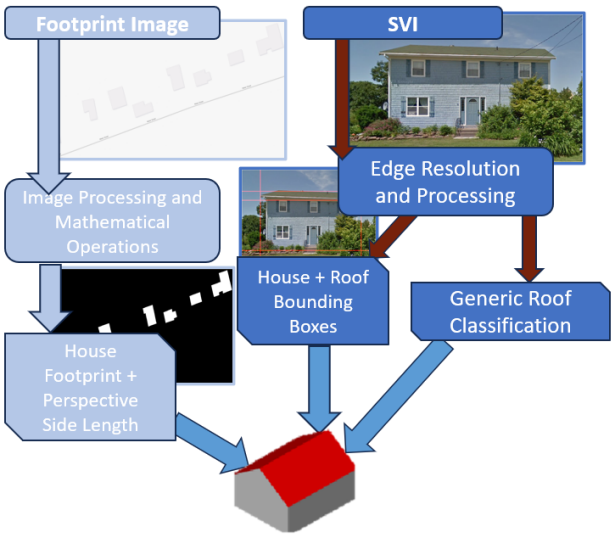

Figure 2: Generation workflow for LOD 2 BIM

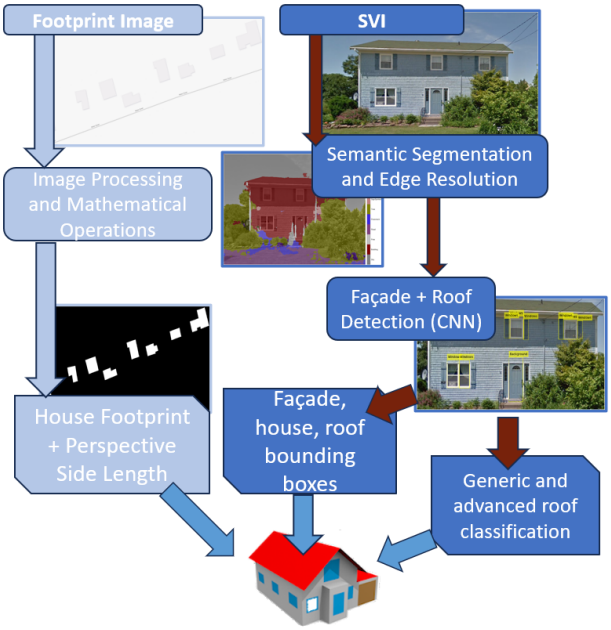

Figure 3: ML supplemented process of generating higher LoD models

bounds of the building of interest. Likewise, if discrete algorithms fail, the outline of the building created from semantic segmentation can be used in place. Line definition can yield simple front or side-facing roofs; however, more complex roof types are too difficult to identify quantitatively. Therefore, after the roof point, top, left, and right of the building are identified, the resulting bounding box should be cropped and classified using a convolutional neural network (CNN) into more complex categories. The classification is then used to map and scale a pre-loaded roof type to the width, height, and footprint specifications of the building. Note that a bounding box is a functional description of an area on an image containing a matrix with the height, width, and bottom left position in indices.

Lastly, the segmented image is cropped to its boundaries for left, right, top, and bottom as well as the building face can be passed through an R-CNN network, that ultimately identifies the windows, doors, and garage features and their scaled locations to be mapped into the overall building model. Images can be extended to the other side faces of the building to provide features on sides besides the front. In applications where these images cannot be retrieved (that is to say, images besides the SVI of the front of the building), a set window-wall ratio can be applied using known architectural patterns based on the north, south, east, and west orientation of the building face. The process augmented with ML to provide higher LODs can be seen in Figure 3.

Development and application of ML

To supplement the overall process, a number of algorithms for morphological and convolutional image processing, and deep learning techniques are applied. Along with image processing techniques and programming, some ML methods must be applied to achieve detail above LOD1. This involves the training procedures, along with the data creation for training and validation. This includes separate training processes and methods for the two different technologies of CNNs and R-CNNs. The computational power of one personal computer on an RTX 3060 graphics unit is utilized for all training and applications. By reducing the input size of the networks, the training methodology can be adopted for lower graphics specifications. The process for the creation of each network involves collecting training data, iterative training cycles, and validation. The workflow for network creation is seen in Figure 4.

Data collection and pre-processing

Training data for facade identification utilizes SVIs from an area that functions as a target case for energy modeling. 442 training images are collected and labeled manually using the MATLAB ground truth data format and built-in tools. The training data is used in the R-CNN architecture, and must be labeled with bounding boxes for

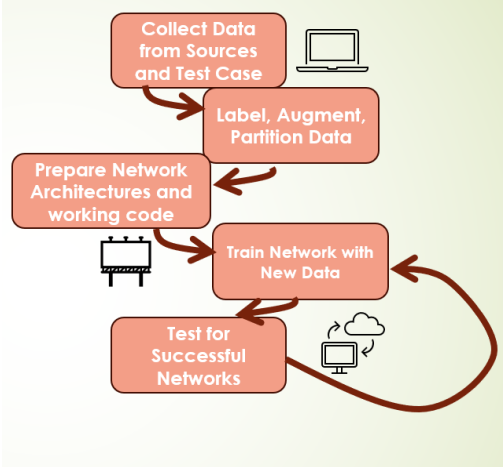

Figure 4: Application and workflow of deep learning networks

each object type of windows, doors, and garages. The labeling process is reflected in Figure 5.

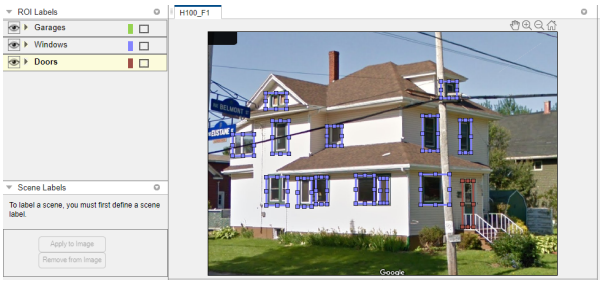

Figure 5: Building labeling process to prepare training dataset

For roof classification, 334 training images are collected of various roof types for the current research work; however depending upon the application, this can be adjusted. The data collection focused on retrieving cropped images containing only the roof of the building that encompassed a wide diversity of roofs. Still, certain architectural patterns are present that bias the commonality of certain roof types over others. The classification categories are derived and sorted by organizing images into file directories with names corresponding to classifications. Originally, twelve complex roof types were sorted; however, the lack of availability for certain examples led to the reformulation of roof types, which is discussed in the training process section.

The data is portioned into training (90%) and validation (10%). Furthermore, the data is augmented and a separate augmented data set is created with randomized hue, brightness, and zoom. This is used as needed to increase available training data for some training iterations. Due to the high computational strain of semantic segmentation training, a pre-trained example is used from MATLAB. The camVid dataset is used to train the semantic segmentation network (Brostow et al., 2008). This network is pulled from a pre-existing MATLAB library already trained for this process.

Training process

Several network architectures are created, trained, and validated for their viability in the proposed methodology, aside from the semantic segmentation network previously mentioned, which is only applied. Network training is optimized to be completed in minimal time using reasonable personal computer specifications. Each network is derived from a known model, trained first with publicly available datasets and then trained on case-specific images, utilizing transfer learning. Note that although they are derived from known architectures, the architectures for each of the networks are customized based on iterative testing during the training process. The collection of more data, and the use of more computational resources can enhance the precision to automate the process fully. All of the custom base CNN architectures created are series networks.

Using an R-CNN network is created for facade identification. MATLAB is chosen for network development due to its widespread application for machine learning, and ease-of-use in training procedures. However, this method is generalized and can be easily implemented using other programming techniques, such as Python. The network architecture is chosen from testing common industry-standard convolutional networks as the base CNN. The *VGG-16* structure is seen to be superior due to its small size and comparable metrics to more convolutionally deep networks, and is used as the basis for the architecture developed. The architecture developed uses an input layer of 256 x 256 x 3, before being passed through a pattern of six blocks that include a convolutional layer with 32 - 3 x 3 filters, which increments the number of filters from 32 to 128, a batch normalization layer, and finally a reLu activation layer, and a 2 x 2 pooling layer to compute features. Borrowing from the architecture of *DeepLabV3*, to increase the effectiveness of bounding box regression, atrous convolutions are applied for the third and fourth

blocks, and a transposed convolution layer, with a similar effect of upscaling the output, is applied in the final layer. The resulting layer structure condenses and upscales the image throughout feature identification providing information based on more overall area and pixels of the image. The final layers consist of a fully connected layer of 256 outputs, followed by two more fully connected layers of 128 and 4 outputs. At this point the four output neurons should contain values relating windows, doors, garages, and background which are evaluated in a softmax layer, and finally a classification layer. For earlier versions of the network, a batch size of 16 is used, to allow it to slowly develop features. In later versions, this is increased up to 128.

The network is trained over several iterations, using differing training settings. Stochastic gradient descent with momentum is applied in the solver, to provide more momentum in reactions from the loss function in the network. This in turn leads to faster convergence. A starting momentum of 0.9 is used for initial training iterations, and for final iterations 0.7 is used due to the network already developing rich feature detection. An initial learn rate of 0.001 is used, decrementing by a magnitude of 10 after every 5 epochs. A total of 100 epochs constitutes a full training session, taking 1-3h depending on the breadth of the dataset. The custom training data collected for this network spanned 442 labelled SVIs, and data augmentation and transfer learning from training iterations with publicly available facade data from Isola et al., 2016, containing urban data with a larger frequency of objects to build foundation.

The roof classification is done through the use of a CNN with a structure similar to the R-CNN base. The roof classification layer outputs a complex roof classification among eight identified types, which are ultimately preloaded sets to be hatched onto the geometric model after classification. Figure 6 reflects the eight classification categories, formed from qualitatively observing the common roof types, and quantitatively classifying the 350 custom training images collected so each category has more than 10 instances. Likewise, some complex roof types are not explicitly used as a classification category, due to a low amount of samples, and instead classified as a complex gabled roof. Note that the networks are only used within the SVI data extraction process. The geometry creation process only incorporates the simple roof types for generation, the more arduous process of hatching complex roofs will be addressed in further work. The networks with capability to identify both simple and advanced roof types have both been developed to supplement the current process and its extension.

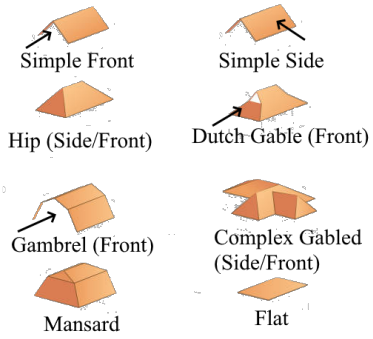

Figure 6: Roof classification used for labeling

Footprint data extraction

As aforementioned the footprint data is extracted from an open street map (OSM) image of a given street. As a web-view screenshot is used, certain steps must be taken to retrieve data in terms of positions in a meter grid. This work has developed the entirety of this process to apply machine vision techniques to retrieve the footprints and recognizes the scaling of the image using OCR technology, and store the resulting information for further use in the proceeding geometry creation. Firstly, a binary image can be created from thresholding the image for known RGB values for buildings and roads. This yields two binary images where pixels not belonging to either roads or buildings are zeros, while the objects of interest are labeled as one. The building and road images are then taken through a set of processes that all involve the use of connected component labeling to define objects. Connected component labeling uses a simple algorithm to identify and group neighbouring pixels, discretizing a binary image into its several objects defined by connected groups of ones, specifically connected horizontally and vertically (4-connectivity) due to the relatively large size of the objects. Simple filtering is done to remove objects with areas that are under 100 pixels in both cases. The building image is then reduced to the outermost boundaries of each region, the building outlines, and the road image is thinned through morphological skeletonization to provide a one pixel wide line throughout the image defining the road.

After creating labelled component images of each

building, the pixel values of the buildings must be converted to real values in meters, by retrieving the pixel scale in the bottom left of this image by cropping the known location of the numbers and passing it through an OCR algorithm. A simple algorithm is written to increment from the consistent pixel location of the left side of the ledger scale, moving one pixel to the right and incrementing a value until it reaches the next fully black pixel. The incremented value is divided by the value retrieved from optical character recognition to provide a ratio for pixels to meters.

The building outline and road images are first cleaned for general noise by filtering region areas under 100, as well as buildings under 20m in perimeter by applying the pixel scale, which are defined as sheds and will not be generated for this analysis. Remaining buildings are indexed for the pixel coordinates of each building. As the program loops through each building, it calculates the road point that is closest to the centroid of the building. This road point, and a road point ten connected points prior and ten connected points further are used to derive the road angle. The individual building coordinates are then rotated by the angle of the road through matrix multiplication. By using the distance between the maximum and minimum x-values, the road perspective side length of the building can be retrieved. This side length is used for several processes including the height formulation from the SVI.

Finally, the last portion of information is a corresponding matrix that identifies the front, left side, right side, and back of the building from the perspective of the road. By using the rotated set of points, a simple method can be used to index the points that are within 0.2m of the maximum and minimum x and y values to define each face. These indices are used to create a matrix that stores values 1 for front face, 2 for left side, 3 for right side, and 4 for back. This matrix is important for the application of facade features later in the process. This matrix has the same length as the footprint point-cloud, and will be "stacked" on itself in further processes that iteratively create the height of the building.

SVI data extraction

After the footprint features are sufficiently expressed in relevant data structures, the SVI must be analyzed. This section will detail the novel developed process of applying the several networks with the addition of discrete image processing techniques, and edge detection adapted from Orhei et al. in 2021 combined with a unique edge processing algorithm that ultimately defines the bounding boxes for the roof, building, and facade features of the building in proportional locations and values that can be correlated to the measurements found in footprint extraction. The SVI is first passed through semantic segmentation to remove common obstructive noise present in suburban areas. This forms a mask that can be used to simplify further tasks, and form a preliminary boundary on the image for the location of the building. Thereafter, to mitigate the amount of noise the mask is dilated by a disk of size 4 to connect any separated building objects, and make sure building edges are within the mask for further analysis.

A novel edge detection method adapted from Orhei et al. in 2021 for building facade identification is used on the masked image. This process (developed by Orhei et al., 2021) utilizes unique edge detection kernels on a grayscaled image to convert the image into a single [nxm] matrix instead of separate colour channels, with a gaussian filter applied to smooth irregularities in the image. After the two prepossessing steps are applied, the image is taken through the process of matrix convolution using the custom kernels G_x, which yields the horizontal lines, and G_y which yields the vertical lines, with an emphasis on having long and connected line results (Orhei et al., 2021). These two results are saved as I_y for vertical edges, and I_x for horizontal edges that are detected. These edge detection kernels reflected in equations (1) and (2), are ultimately used to retrieve prominent horizontal and vertical lines which are combined using equation (3). Note that the proposed methodology utilizes a median filter instead of a gaussian filter for the smoothing step, and utilizes the same kernels and overall process.

$$Gx = \frac{\begin{bmatrix} -2 & -2 & -8 & -2 & -2 \\ -1 & -1 & -4 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 4 & 1 & 1 \\ 2 & 2 & 8 & 2 & 2 \end{bmatrix}}{512} \tag{1}$$

$$Gy = \frac{\begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -8 & -4 & 0 & 4 & 8 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}}{512} \tag{2}$$

$$\sqrt{I_x^2 + I_y^2} = I_e \tag{3}$$

Then, each of the edge images are thresholded for the most prominent lines (values above 1) dilated, and binary skeletonization is applied. This is done to find the most prominent lines, such as the division between the sky and roof, and the building edges. The dilation before skeletonization is used to connect detached lines that belong to the same feature to increase line length. Lines that are less than twenty-percent of the total building width and building height are taken out of each image. The remaining lines are passed through an algorithm that loops through the pixels belonging to each line object, computing the angle between it and the previous and further points to it (a spacing of 11 is used for this). This is done to assign a value to each pixel that corresponds to its given angle, helping identify straight lines. Using this method, lines can be filtered further for their local angles; this removes spurs and other features within the lines. Likewise, the most prominent vertical lines (70-90 degrees), horizontal lines (0-20 degrees), and slanted lines (20-70 degrees) are saved for further use.

An additional skeletonized image is produced from binary skeletonization of I_e, without prominent lines threshold. This provides an overall skeleton that is used to regress the boundaries of the building to more accurate values. A discrete algorithm then uses the defined slanted, vertical, and horizontal prominent lines along with the overall skeleton to define certain parameters. First, it looks for the largest prominent line at the highest points of the image, the angle of this line, whether it is horizontal or slanted, is used to define whether the generic roof type is a front-facing or side-facing roof. If this is found, the new top boundary is set to the highest point of these lines, and the roof starting height is defined as the bottom of this line if it is a slanted line. Likewise, if it is a slanted line at the top, then the building is a front-facing roof. If a horizontal line is found at the top, it defines the building as a side-facing roof and attempts to find the second major horizontal line under this line, which is defined as the starting point of the roof. Then, vertical lines close to the existing building edges (within 60 pixels) are found, and the leftmost and rightmost lines are used to define the respective boundaries. If any of the information is not found, the boundaries of the overall skeleton are used, and if the roof point is not found, a roof point is defined as 1/3 from the top of the building.

Finally, the bounding boxes created for the building and roof are cropped and passed through the CNN (the roof) and the R-CNN (building face) for advanced roof classification and facade detection. The resulting bounding boxes for the windows, doors, and garages of the cropped building image, and the advanced roof classification are saved in corresponding matrices for each building. In this way, the primary outputs of this process are bounding boxes for the building and roof, as well as the generic (side-front) and advanced roof classification and bounding boxes for the windows, doors, and garages for each building. These bounding boxes will define the ratio between the height and width of the building, and the ratio of the roof to the entire building height, along with the location and proportional size of the buildings, that can be normalized with the perspective side length found in the footprint extraction process. The overall process for SVI data extraction is shown in Figure 7.

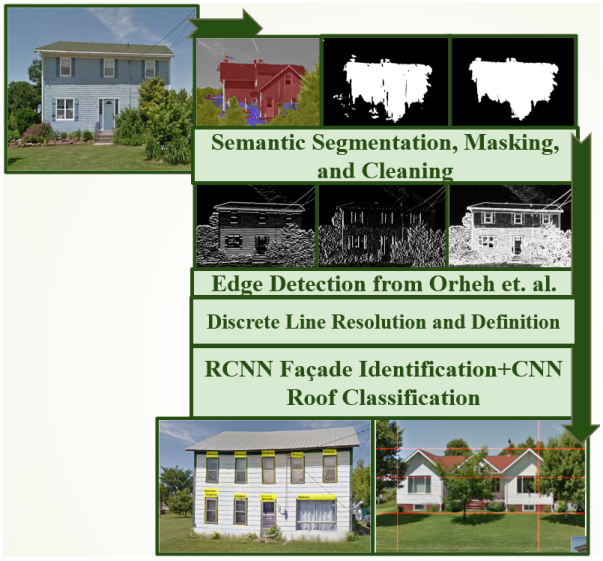

Figure 7: SVI data extraction process

Geometry creation

This work outlines a developed method for applying the proportional data from the SVI and the discrete measurements from the OSM image using simple geometric layering techniques for roof generation to generate point-cloud data compatible with applications like *Rhino-Grasshopper*. Firstly, the perspective building side length is used to generate the height of the building, using the height-width ratio from SVI data extraction. This height is incrementally generated with an arbitrary number of steps (48 in this study) to form the frame of the building by stacking the footprint x-y coordinates with incrementing z. Thereafter, the roof type

derivation can be used after the z value reaches the roof point. From this point, another 24 points are generated, and the remaining height is generated with two simple methods for front-facing or side-facing roofs. For front-facing roofs, a simple gable geometry is created by first deriving the angle of the building, allowing for the program to simply rotate the building similar to the method for perspective road length, and incrementally move values to the center by adding half of the full length divided by 24. The points that must be moved can be defined by using a maximum and minimum x value which is also incrementally being moved towards the center. This process utilizes the same method with y values for the side roofs. This generation process can be extended using mathematical patterns in other roof types, or simply by warping an existing point-cloud of the roof type to fit the given building footprint. The created process focused on the proof of the separate components, and due to time constraints cases for complex roof types are not made. The classification results are used to fit LOD2 general roof shapes onto the model. The point-cloud is created with a corresponding color matrix, which is used to store data on the type of surface. Roofs are coloured slightly darker than the buildings they belong to, using random generation to define building colour. The bounding boxes from the cropped SVI are used in conjunction with the point-cloud for each building to map the facades onto the building faces. This can be done by simply indexing the points that are between the x-z (for the front face y-z for the side faces) that are within the bounding boxes. The bounding box values have their height and width scaled to the pixel value by taking the frame width of the cropped image and dividing it by the building side length. Likewise, the x-y coordinates for the building are saved separately and subtracted by the minimum x value to retrieve values starting at zero. In this way, the coordinates in the bounding boxes can be made to correspond to points within the matrix, and the colour can be saved for each of the different facade features to be separate in the matrix. This will, in turn, create a building with information above LOD2.

Results

The work will attempt to test the developed networks (with the semantic segmentation network neglected as it is pulled from other sources), and the overall developed methodology in its ability to create volume-accurate building geometry. Likewise, training, testing, and validation are done only on single-family detached sub-urban buildings, as this is the intended use-case for the process. To evaluate R-CNN networks average precision is used. The mean-average precision is computed as the average precision across all recalls, the average of the integrated area of the precision with respect to the recall, incorporating the area-over-union between the result and the true box. In this way one can quantify both precision and recall with one metric. Precision refers to the amount of true positives compared to the amount of true and false positives in a network. Recall refers to the number of true positives compared to the number of true positives and false negatives. These two metrics define how much an object detection network falsely identifies something (recall), and missing an identification (precision). A false negative (FN) refers to missed detections, a true positive (TP) refers to the correct detections, false positive (FP) refers to incorrectly identified objects.

Network Validation

Each network is trained in several iterations, with successful network branches. After convergence is noted in a network model, training is stopped, and the network is tested. The most successful networks are carried to different iterations. The convergence for the networks is observed to take around 3 hours. The R-CNN network is trained over 4 iterations, while the base CNN network took significantly less time and completed all training until convergence for sessions within 3 hours and produced a viable result. All precision values are normalized from 0-1 and evaluated on 10% of the (unused) training dataset containing 204 windows, 31 doors, and 1 garage.

The R-CNN network had several objects that it is to detect. As the number of windows far outweighs doors (8:1) in the validation and training datasets, a combination of the mean precision values over all object types will sparsely quantify the division between certain objects. For this purpose, the mean precision is considered for the most prevalent object type, windows, to prove the viability of the network. It is expected to have a significant difference between precision values over different objects due to vastly different training quantities for each object type. Table 1 reflects the mean-average-precision (γ) along with the simple average precision (ζ) (the sum of all precision divided by the number of precision values), and the training time (τ). It is found that the precision of the most well-performing network reached 0.70, with the mean precision over all identifications reaching 0.93 for windows. The precision for doors is significantly lower (0.27), and it cannot find the one garage example resulting in a precision of

Table 1: R-CNN training and validation insights

τ	γ	ζ
12h	0.70	0.93

zero. This results from the training data containing 25 times more examples of windows than doors, and 40 times more than garages. Other networks that performed worse for overall values, due to lower window precision, had higher precision for doors (0.41). The number of garage examples is seen to be far too small to form a feature evaluation in the networks. Qualitatively, some features are noticed from facade identification. Due to the high amount of screen doors in the testing and target cases, the identification would often identify these doors incorrectly as windows, often overlapping a window and door identification on top of a door. This is addressed in the current process when an overlapping window and door are detected to replace the classification with only a door. Most doors in the use-case are screen doors including a window within it; as such, further work will incorporate such screen doors as a separate category. It would also have trouble with shaded building faces and particularly dark windows. These errors are reflected in Figure 8. The CNN network is successful in its

Figure 8: Common errors in facade identification

classification task with a mean accuracy (β) of 0.79 overall classifications, and an accuracy, ρ, of 0.81 for its most popular case (simple gable side-roofs), reflected in Table 2. 4 iterations of training are used, and data augmentation is applied to double the data size for the first iterations. The training data is reduced to only real images for the second two iterations, which incremented batch size and decreased the starting learn rate to 0.0001.

Table 2: CNN results with precisions from 0-1

τ	β	ρ
3h	0.79	0.81

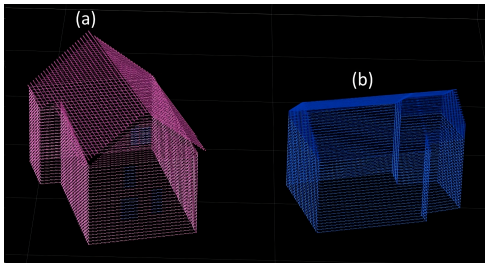

Figure 9: Resultant point-cloud of process (a) front-facing roof, (b) side-facing roof

Geometric results

The geometric results created are compared to the geometry extracted from taking measurements on site and using modeling tools within *Grasshopper* processes described prior. Table 3 reflects the results, V represents the volume (m^3), with subscripts 'xyz' and 'M' representing this generation process and manual geometry creation, respectively. Δ reflects the percent error with subscript 'fp' representing footprint (in m^2).

Table 3: Geometry comparison of generated building 3D models with manual models

House No	$V_M(m^3)$	$V_{xyz}(m^3)$	Δ_v	Δ_{fp}
1	297.6	290.7	-2.31%	-1.36%
2	386.5	388.7	0.58%	-3.30%
3	386.9	373.8	-3.39%	-3.33%
4	370.2	369.4	-0.19%	-7.34%
5	751.8	794.6	5.70%	-7.15%

The area and volume of both files are within 95% of each other on average; however, the XYZ coordinates have larger footprints and volumes. There is an outliers where the inaccuracy is around 10%; in these cases, the manual data, both the footprint (3-5%) and the volume (9-11%) deviate more compared to the other samples. Figure 9 reflects the output result of the methodology.

Discussion

Overall, the results seem to show that this process is viable, although there is room for further refinement in several avenues. Further work to refine this method for its intended purpose in modeling a target sub-urban area

should be pursued.

Machine learning application

Although the precision and recall results proved the viability of the facade identification, poor performance is observed on some objects. It is clear that performance is greatly affected by the number of objects in the training sample. Nonetheless, both convergence and a relevant result are found from restricted computational resources, data, in a reasonable amount of training time. Expansion of the dataset is required to predict rare cases accurately. Yolov4 and Yolov5 architecture should be explored as other work on facade detection by Sezen et al., 2022 implies these are superior architectures for facade identification. In this study, the computational resources required to estimate anchor boxes, and have reasonable size specifications for this network proved too consuming. Optimizations can be made in the architecture, or a simpler existing architecture can be borrowed for the base CNN.

The classification network is also relatively successful, with an overall accuracy of 0.79. Upon closer inspection, the network fails the majority of the time when encountering an uncommon roof type. Roof types like mansard had less than 10 training samples, and definite features could not be developed. The same error due to a lack of diverse data is observed for this network. The CNN network can be improved by expanding the breadth of the dataset, particularly focusing on uncommon roof types. However, overall, the proposed generalized method remains the same.

Further optimization can be pursued by creating a larger base for transfer learning by utilizing more data sources for the networks. Unfortunately, too much urban data caused the network to develop noticeable patterns that are defective for the sub-urban environment, and the generally lower proportion of window boxes caused the discrepancy. Likewise, facade information, including garages, is particularly sparse. There are also a number of prepossessing techniques that could be applied to enhance window features before the object detection process, such as edge enhancement. Although improvements can be made on algorithm choice, this study proves that such application of object detection networks can provide a robust methodology for automating information beyond LOD2.

Geometry creation

The geometry creation encountered some errors as well. The average deviation (2.6%) for the footprint data must be considered first. The OSM data is seen to provide different footprints when compared to the manual data. This, in turn, accentuates the volume discrepancy between the two models. The footprint can be seen to contribute significant portion of the error, which reduces all volume errors under 5% compared to manual when neglected. Likewise, although the CNN is developed, the geometry hatching for complex roof models cannot be mathematically defined for geometry creation with diverse footprints due to time constraints. The development of this model would allow for far greater accuracy due to the known success of the classification network itself.

Another portion of the error likely comes from the method used to retrieve the height of the model. A more robust method shown in equation (4) (Zhao et al., 2019), computes the height, h, based on the pixel length h_b of the building above the center vertical center line of the image, the focal length of the camera f, d is the distance between the camera and the building, h is the estimated building height, and h_c is the height of the camera.

$$h = \frac{h_b \cdot d}{f} + h_c \tag{4}$$

In this study, there is no access to camera specifications for the perspective pixel length, which is required for the building-image height in meters h_b. Accurate information about the exact position when the image is taken is unavailable. This process assumes that the camera is perfectly planar to the building, which may not be true.

Remarks on general process

Additionally, it should be noted that two main data outputs from the footprint extraction and SVI analysis can be stored intermediate to geometry creation. A user can review and tweak bounding box results to account for failures and uncommon features in the SVI. This would allow the generation process to be semi-automated, with the geometry creation aspect being completely automated after the set of bounding boxes and identifications for SVIs are tweaked. In this way the tedious portion of geometry creation can be reduced to a menial review of a set of SVIs for which the results are already 70-90% complete, and the rest of the process is automated. The current process is fully automated (2 clicks) after file directories for SVIs have been defined, and gener-

ates a point-cloud to be used in further programs like *Rhino-Grasshopper* for BEM generation. Furthermore, the entirety of the process for an existing building takes less than 2 minutes to fully complete on an RTX3060 graphics card. The process can be set to run without supervision. Thus, the process is deemed rapid as it deals with the time complexity of geometry generation through improving automation (reducing the number of clicks), consequently removing the need for manual work hours per existing building (time complexity). Clearly this method is best framed with an easy-to-follow guided user interface to further streamline application.

Limitations and further work

The developed process addresses a specific use-case for sub-urban buildings composed almost entirely of single-family buildings; several limitations are attributed to this process. Connected buildings are difficult to address, the edge identification and segmentation methods will incorporate noise from connected buildings. In these cases, however, the connected mass can be treated as one full building. There are no examples of connected buildings in the use case or the dataset. Likewise, urban buildings that are not fully captured in SVI images, that is to say, tall or extremely wide buildings, will not be addressed with this process as the proportion-based height retrieval will be difficult, especially if significant perspective distortion is incorporated by pointing the camera upwards (therefore making the height pixels not proportional due to the pitch of the camera). Lastly, it remains difficult to incorporate accurate roof forms within this process. The building must be fully encapsulated in the SVI, and vegetative noise will be automatically filtered. Still, if vegitative noise completely obscures a large portion of the building pivotal lines may be missed.

Further research work before the application of this process is to create hatching mechanisms for more advanced roof types, which are currently being classified but not applied. Likewise, instead of processing an OSM image, the method could directly pull data from the OSM API, which is also free. Cloud computing could also be applied to significantly enhance network training for custom networks, and make it viable to train a custom semantic segmentation network. Additionally, the packaging of this program into a user-oriented design with an interface is not yet pursued. Moreover, future work may also include refinement of the specific networks used. This work serves to solidify the feasibility of using these modern techniques for the purpose of BIM modeling. This methodology outlines the processes required to relate these several algorithms to create a point-cloud model efficiently. The building point-cloud models can then be incorporated using third-party software such as *Rhino-Grasshopper* to tune them, separate the facade features and buildings (generating them separately), and apply the R-values and other parameters to create BIM models for BEM. Likewise, as stated previously, the application of this methodology is specifically for sub-urban buildings, where the vast majority of the training and validation cases are single-family detached buildings. To extend this process to a larger variety of home types, more work must be done in generalizing the process, along with collecting and applying training data from other home archetypes. The screen doors resulting in overlapping window and door classifications will be added a separate category for facade detection.

Conclusion

In this work, a machine learning (ML) based method is proposed to rapidly generate building information models for energy modeling. Various ML algorithms are integrated to generate 3D building models for various levels of detail (1-3) using OpenStreetMaps and street view images. The developed network models can supplement the footprint data to provide a relevant geometric model, with mapped window, door, and garage features. The training results indicate that this method can provide viable results with a relatively low amount of training samples. Expansion of the dataset is required to predict rare cases accurately. The facade detection is seen to have an accuracy of 71%, with roof classification being at 79%. Network architecture choice, increasing the amount of training data, and utilizing cloud-computing are all methods that would improve the detection. The overall geometry modeling accuracy is found to be 95% compared to manual data. It is noted that user assistance can be incorporated to adjust bounding boxes at every step to maintain most of the automation, reducing the time complexity of the workflow greatly while allowing for much higher overall precision. The real value of the time complexity of this work can be quantified in requiring under 10 clicks (including those clicks for setup) after a set of files for the SVIs, and a given footprint image is provided, compared to the hundreds of clicks required to create geometry manually. The point-cloud output of this process is easily integrated to existing workflows in programs such as *Rhino-Grasshopper* to create BEMs, which is part of the ongoing research.

Acknowledgements

Funding received by Dr. Kuljeet Singh Grewal from the Natural Sciences and Engineering Research Council of Canada (NSERC) under the Discovery Program is duly acknowledged.

Nomenclature

CNN	Convolutional Neural Network
R-CNN	Regional-Convolutional-Neural-Network
Yolo	You Only Look Once object detector (v3-v5)
SVI	Street-View Image
γ	Classification and object detector mean average precision
Δ	Difference in percent
LiDAR	Light Detection and Ranging
G_x	Convolutional kernel for edge detection (horizontal)
G_y	Convolutional kernel for edge detection (vertical)
I	Refers to [nxmx3] image
ζ	Average object detector precision on populous case
ρ	Average classification accuracy on populous case
UAV	Unmanned Aerial Vehicle

References

Al-Habashna, Ala'a. "Building Height Estimation Using Street-View Images, Deep-Learning, Contour Processing, and Geospatial Data." 2021 18th Conference on Robots and Vision (CRV), 2021.

Brostow, Gabriel J., Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. "Segmentation and Recognition Using Structure from Motion Point Clouds." Lecture Notes in Computer Science, 2008, 44–57.

Dai, Menglin, Wil O.C. Ward, Gregory Meyers, Danielle Densley Tingley, and Martin Mayfield. "Residential Building Facade Segmentation in the Urban Environment." Building and Environment 199 (2021): 107.

Gil, Alejandro, Laia Núñez-Casillas, Martin Isenburg , and Alfonso Alonso-Benito. "A Comparison between LiDAR and Photogrammetry Digital Terrain Models in a Forest Area on Tenerife Island." Canadian Journal of Remote Sensing 39, no. 5 (October 2013): 396–409.

Gu, Jiuxiang, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, et al. "Recent Advances in Convolutional Neural Networks." Pattern Recognition 77 (2018): 354–377.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-Image Translation with Conditional Adversarial Networks." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

Jaeger, Ina, Glenn Reynders, Yixiao Ma, and Dirk Saelens. "Impact of Building Geometry Description within District Energy Simulations." Energy 158 (2018): 1060–69.

Maset, E., A. Fusiello, F. Crosilla, R. Toldo, and D. Zorzetto. "Photogrammetric 3D Building Reconstruction from Thermal Images." ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W3 (2017): 25–32.

OpenStreetMap. Accessed November 13, 2023. https://www.openstreetmap.org/.

Orhei, Ciprian, Silviu Vert, and Radu Vasiu. "A Novel Edge Detection Operator for Identifying Buildings in Augmented Reality Applications." Communications in Computer and Information Science, June 2021, 208–19.

O'Donnell, James, Linh Truong-Hong, Niamh Boyle, Edward Corry, Jun Cao, and Debra F. Laefer. "Lidar Point-Cloud Mapping of Building Façades for Build -ing Energy Performance Simulation." Automation in Construction 107 (2019): 102905.

Pang, Hui En, and Filip Biljecki. "3D Building Reconstruction from Single Street View Images Using Deep Learning." International Journal of Applied Earth Observation and Geoinformation 112 (2022): 102859.

Sezen, G., M. Cakir, M. E. Atik, and Z. Duran. "Deep Learning-Based Door and Window Detection from Building Façade." The International Archives of the Photogrammetry, Remote Sensing Spatial Information Sciences XLIII -B4-2022 (2022): 16–20.

Zhao, Yunxiang, Jianzhong Qi, and Rui Zhang. "CBHE: Corner-Based Building Height Estimation for Complex Street Scene Images." The World Wide Web Conference, 2019.