Computer-Aided Design and Applications
© 2008 CAD Solutions, LLC
http://www.cadanda.com

Automatic Model Simplification and Reconstruction from Geographic Information System Data for Computer-Aided Engineering

Chiet Sing Chong[1] and Yi Su[2]

[1]Institute of High Performance Computing, chongcs@ihpc.a-star.edu.sg
[2]Institute of High Performance Computing, suyi@ihpc.a-star.edu.sg

ABSTRACT

This paper describes an automatic reconstruction methodology for the creation of 3D models suitable for Computational Fluid Dynamics (CFD) simulation. The objective is to design and develop an automatic model reconstruction methodology to convert Geographic Information System (GIS) data to CAD data which can be used directly for finite element mesh generation. This will equip users with a reliable and effective tool for the rapid reconstruction of geometrical models which satisfy their computational requirements and simulation contraints. Specifically, the reconstruction algorithm will be able to automatically process building data in the ESRI shapefile format, suppress unnecessarily detailed features, resolve non-manifold topology, cluster buildings which are close together and export the 3D model into a universal exchange format suitable for the intended CFD application.

Keywords: automatic reconstruction, feature suppression, image processing, GIS, CAE.
DOI: 10.3722/cadaps.2008.296-306

1. INTRODUCTION

In the typical computer-aided engineering workflow, manual repair and manipulation of the geometrical model is often required to obtain a version that meets the requirements of the finite element mesh generation process. A mesh-friendly geometrical configuration will ensure the success of the automatic mesh generation procedure and maintain acceptable mesh resolution and element quality. These are critical factors for the achievement of reliable simulation results while maintaining an affordable computational cost. Unfortunately, to obtain a mesh-friendly model is often a bottleneck in the computer-aided engineering workflow due to its highly manual nature. This poses a huge hurdle in shortening the simulation turn-around time, especially when there are many geometrical entities involved, as in the case of modeling urban buildup.

Building information in an urban buildup is often encapsulated in GIS data. This information include the profile or footprint of the building, together with the building height, as well as a host of other related data. The GIS is a system for storing, analyzing and managing data and associated attributes which are geographically-referenced. In the industry, some examples of major commercial offerings are by ESRI [1] and Mapinfo [2]. Government and defense organizations often use custom software, open source products, such as GRASS [3], or more specialized products that meet a well defined need.

With the maturing of CAE techniques, GIS data has also been used for a wide range of scientific applications, including environmental impact assessment, pollution prediction, hazard forecasting, urban traffic simulation and infectious disease modeling. Chu *et. al.* [4] performed CFD simulations to determine dispersions of emissions from vehicles traversing the streets. The CFX-Build component of CFX5.5 [5] is used to construct the geometry for simulations. The work by Coirier and Kim [6] focused on the development of a CFD-Urban modeling tool for modeling contaminant transport and dispersion. Baffour *et al* [7] proposed a methodology to convert multi agency GIS data into a citywide 3D model to support chemical and biological dispersion simulation. Models presented here includes using remote sensing based building and cartographic morphology data, and GIS shape file building data in the form of building foot and top prints.

1.1 Simulation Modeling using GIS Data

Computational analyses, such as CFD simulations, using GIS data as a starting point is becoming more commonplace. Unfortunately, GIS data is not immediately useable for the creation of CAD models suitable for mesh generation and simulation. The CAD models generated directly from GIS data often requires extensive manual repair and manipulation to obtain a valid version. Moreover, the shape of the buildings contained in the GIS data can be very complex or contain too many details which are unnecessary in the simulation process. This is especially the case in the context of a large scale CFD simulation which involves hundreds or thousands of buildings over a huge domain.

With regards to the simplification and repair for such GIS data, some state-of-the-art are discussed as follows. Sugihara [8] presented an algorithm for filtering out noisy edges and breaking down complicated polygons into simpler primitive shapes, which can be used to generate 3D models for urban planning. The work by Baker and Semwal [9] focused on the 3D visualization of topographic data available from maps. The idea was to first extract a two dimensional rectangular region from the map, and then use contour line extraction for the visualization.

Glander and Dollner [10] proposed a cell-based generalization of 3D building groups, with each cell structure consisting of a cluster of buildings with weighted-average height. Anders [11] proposed a typification and aggregation approach for building clustering and simplification. However, this approach is not suitable for handling buildings with highly complex shape. Forberg [12-13] presented an alternative approach for building simplification and clustering suitable for orthogonal building structures. It is a promising approach but the effectiveness is dependent on the constraint of orthogonality, which is not the case in many actual cases.

In this work, the authors focus on reducing the simulation turn-around time by replacing the manual task of model rectification and simplification with an automatic methodology. This is especially important for large-scale simulation where a large amount of GIS data is involved. The objective is to design and develop an automatic algorithm to perform rapid conversion and simplification of Geographic Information System (GIS) data to a mesh-friendly CAD model to speedup the pre-processing stage of the computational analysis. The key functions are the suppression of excessively detailed building features, the repair of faulty or degenerate geometries and the clustering of buildings which are closed together.

2. METHODOLOGY

This section describes the methodology behind the automatic simplification and reconstruction algorithm. The input GIS data is based on the ESRI shapefile specification [14]. Depending on the way the GIS data has been sampled, a typical ESRI shapefile might contain numerous geometric entities with configurations that might pose problems to finite element mesh generation. These problematic configurations include zero-length edges, edges with extremely acute angle at the vertex and edges with small gaps or overlaps. In addition, some buildings contain unnecessarily detailed features with respect to the desired finite element size. These detailed features need to be suppressed so that the number of finite elements in the resulting mesh is not overly dense. Moreover, buildings which are in close proximity can be clustered together to further reduce the complexity of the simulation.

The overview of our proposed algorithm is illustrated in Fig. 1. The algorithm is a hybrid of vector-based and image-based operations. In general, vector-based operations require much less computational overhead as compared to image-based operations. However, the image-based operations provide an elegant way of resolving some difficult geometrical issues, which we will elaborate in the subsequent sections. The algorithm involves 3 stages:

- Suppression – This is a vector-based process. The objective is to remove entities which are deemed too small to make any significant impact on the simulation.
- Simplification – This is an image-based process. The objective is to reduce the complexity of the geometry by removing detailed features and clustering entities which are in close proximity to one another.
- Repair – This is a vector-based process. The objective is to resolve geometrical configuration that might result in non-conformity in the resultant mesh.

User inputs that are required in the whole process-flow are listed as follows:

For vector-based suppression processes:

a. Small building's Tolerance (area), B_a

b. Low Building's Tolerance (height), B_h

For image-based simplification processes:

c. Pixel dimension (pixel per unit area), *PPA*
PPA represents the size of a pixel in the map. A small *PPA* value maintains the original shape of the buildings during the image-based processing, but a large *PPA* value will enable users to simplify the building shapes to a larger extent.

d. Building's clustering tolerance (gap tolerance), T_c
This gap tolerance will determine the size of the structuring elements, which is $T_c/2$, in the image closing process. In this process, small gaps between buildings that are unnecessary for the simulation process will be closed up. It will also cause buildings to cluster together if they are close to one another and may eventually merge together if they have similar heights.

e. Building's feature simplification tolerance (height tolerance), T_f
This feature simplification tolerance will decide the size of the structuring elements, which is $T_c/2$, in the image opening process. In this process, uneven edges and tiny fin-like structures will be removed.

f. Building's merging tolerance (height tolerance), T_m
When two connecting buildings are within the merging tolerance, T_m, they will be merged into a single building with one uniform height, *H*. Here, *H* is the area-weighted resultant height given by

$$H = \frac{A_1 H_1 + A_2 H_2}{A_1 + A_2} \tag{2.1}$$

where A_1 and A_2 are the floor areas, and H_1 and H_2 are the building heights of the two combining building blocks, respectively.

For vector-based repair and clean-up:

g. Gap-repair Tolerance, T_{gap}
This gap-repair tolerance is used in the subsequent vertex-to vertex and vertex-to-edge snapping during the geometries clean-up process.

2.1 Automatic Suppression of Geometric Entities

The suppression operation suppresses buildings based on geometrical tolerances. These tolerances include area-based tolerances and height-based tolerances. These functions are useful to remove small entities, like water tanks situated on top of the building, as well as low buildings which do not have significant impact on the results of the simulation. These suppression operations will require user's inputs such as height and area tolerances as well as meshing conditions, such as the minimum element size. The height field values can be extracted from the GIS data readily. Note that the suppression of any features or buildings is subjected to a conjoined metric of height and area, and the subsequent processes, e.g. the type of simulation performed on the generated CAD models. For example, a very low building with a big floor area may be significant in subsequent, says, CFD smoke simulation.

2.2 Automatic Simplification and Clustering of Complex Shapes

To reduce the simulation to a tractable scale, the geometric model needs to be simplified. Ideally, buildings with complex shapes need to be replaced with a simpler representation so that the mesh size of the overall model can be reduced. For a large urban build-up area, manual simplification will take a very long time. To address this issue, an automatic simplification module has been developed.

As contrasted to the classical polygonal simplification algorithm proposed by Douglas and Peucker [15], our method is based on an image morphology approach where the polygonal data of the region-of-interest is first digitalized or rasterized into a single image. This is done by mapping the GIS polygons over a pixel-grid of the user's required resolution. This resolution decides the fidelity of the building shapes on the image with the actual shapes from the GIS building geometries. This image will undergo segmentation process and the interior of a building will be filled with unique image representations given by the building ID and the height value of this building. To simplify the shape of the polygon, this image undergoes a series of image morphological operations. The resultant image represents a simplified profile of the polygon. Finally, the image undergoes a contour tracing operation [16] to recover the simplified polygonal data.

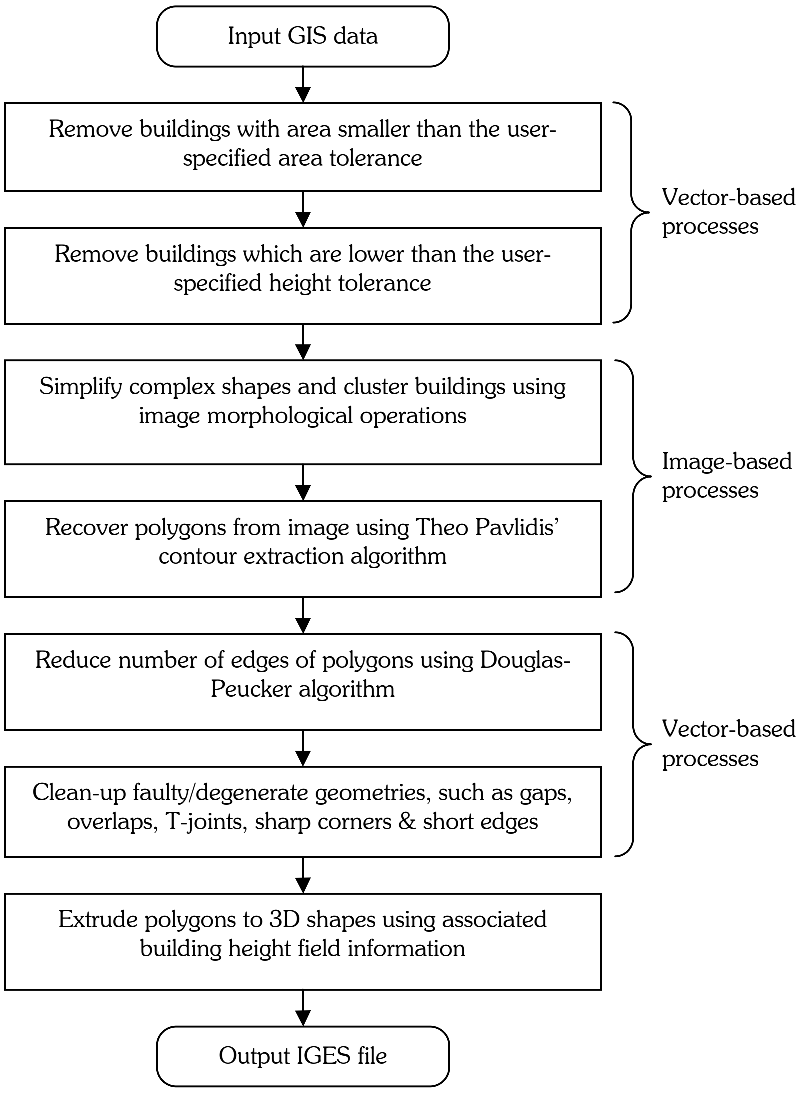

Input GIS data
Remove buildings with area smaller than the user-specified area tolerance
Remove buildings which are lower than the user-specified height tolerance
Vector-based processes
Simplify complex shapes and cluster buildings using image morphological operations
Recover polygons from image using Theo Pavlidis' contour extraction algorithm
Image-based processes
Reduce number of edges of polygons using Douglas-Peucker algorithm
Clean-up faulty/degenerate geometries, such as gaps, overlaps, T-joints, sharp corners & short edges
Vector-based processes
Extrude polygons to 3D shapes using associated building height field information
Output IGES file

Fig. 1: Overview of the algorithm.

The basic morphological operations [17] used in the process: image dilation and erosion. Image dilation is basically a controlled expansion of an object on an image, while image erosion is a controlled shrinkage of an object on an image. Combinations of image dilation and erosion operations offer the two main morphological operations: image opening and image closing. In the automatic simplification processes, image opening is employed to smooth the contour of the objects, thus breaking narrow isthmuses and eliminating thin protrusions. Image closing, on the other hand, is employed to fuse narrow breaks and long thin gulfs, eliminate small holes, and fill gaps between two objects. The Theo Pavlidis' Algorithm [15] is employed in the process of contour tracing to extract the boundary of the building from the image. The simplification process based on image morphology is illustrated in Fig. 3. Some analogies between image-processing and polygonal modeling:

- Noise reduction on images ≡ suppression of small buildings
- Image thresholding ≡ removal of low buildings

- Varied combinations of image dilation & erosion ≡ building simplification & clustering
- Image segmentation ≡ building differentiation

A is a set of pixels on an image. *B* is a structuring element with 3-by-3 matrix. Φ denotes an empty set.

The dilation of *A* by *B* is the set of all *x* displacements such that $\hat{B}$ (the reflection of *B* about the origin of the set) and *A* overlap by at least one non-zero element and is defined as

$$A + B = \{x \mid (\hat{B})_x \cap A \neq \Phi\} \tag{2.2}$$

The erosion of *A* by *B* is the sets of all pixels *x* such that *B*, translated by *x*, is contained in *A* and is given by

$$A - B = \{x \mid (B)_x \subseteq A\} \tag{2.3}$$

The opening of *A* by *B* is simply the erosion of *A* by *B*, followed by a dilation of the result by *B* and is given by

$$A \circ B = (A - B) + B \tag{2.4}$$

The closing of *A* by *B* is simply the dilation of *A* by *B*, followed by a erosion of the result by *B* and is given by

$$A \bullet B = (A + B) - B \tag{2.5}$$

In image processing [17], the opening operation generally smoothes the contour of an image, breaks narrow isthmuses, and eliminates thin protrusions. On the other hand, the closing operation tends fuse narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour. The metrics of the height value and area value do play important roles during the imaging operations. Small buildings with good heights will not be removed during image opening unless the user specified. Similarly, connected buildings with a large height difference will not be merged.

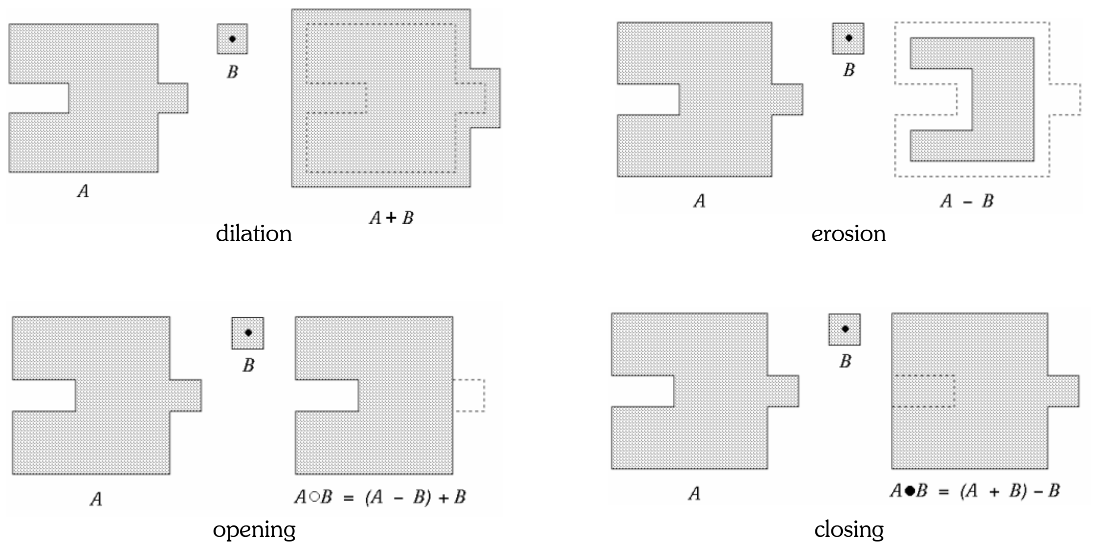


Fig. 2: Different image morphological operations.

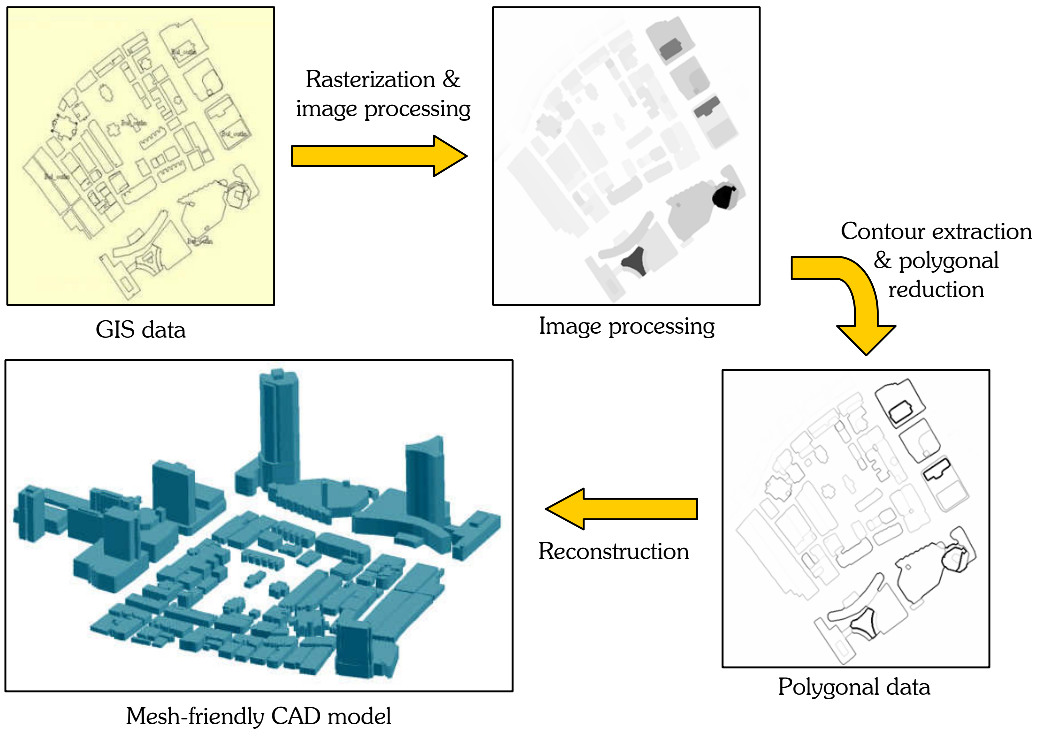


Fig. 3: Simplification process based on image morphology.

Apart from simplifying individual building, the simplification process can also be applied to a collection of buildings. By grouping a set of buildings and converting them into one image, the same image morphology can be used to cluster buildings which are in close proximity to one another. This will further reduce the complexity of the model and hence, reduce the number of elements in the subsequent finite element mesh.

2.3 Automatic Repair of Faulty/Degenerate Building Geometries

The shapefile data usually contains entities which have small geometrical gaps or overlaps, as illustrated in Fig. 1. These geometrical kinks are the result of inaccurate sampling or data acquisition and can cause severe problems while performing 3D Boolean operations prior to mesh generation. The small gaps or overlaps will manifest as sliver surfaces which force the mesh sizes to be extremely small or degenerate around their vicinity. To resolve this problem, an automatic gap closing/repair function is implemented to snap the buildings to each other based on a user-defined tolerance. This will free the user from having to repair the model manually, which is a tedious and time-consuming process. After the gap closing/repair function, it is necessary to resolve the T-joint geometrical configuration, as this configuration will result in a non-conforming finite element mesh.

The gap closing/repair function involves vertex-to-vertex and vertex-to-edge snapping processes:

i. Vertex-to vertex snapping: A pair of vertices (each lying on a different polygon but are in close proximity to one another) are merged if they are within a user-specified tolerance. In our practice, the mid-point of the vertex pair to be the merging point.

ii. Vertex-to-edge snapping: When a vertex of a polygon is within the user's specified gap closing tolerance with an edge from a neighboring polygon, this vertex will be snapped onto this edge, as illustrate in Fig. 4.

A vertex-to-edge snapping will result in a non-manifold edge topology, as illustrated in Fig. 5. The handling of non-manifold edge topology is achieved by a vertex insertion and edge splitting algorithm to realize a conforming vertex-connection along the T-joint. This process is important for connecting buildings as it will subsequently affect the mesh generation and mesh conformity of the finite element model created at a later stage. Any edge of a polygon (building) that is connected to another edge of different polygon (building) will have to ensure vertex conformity along the edges.

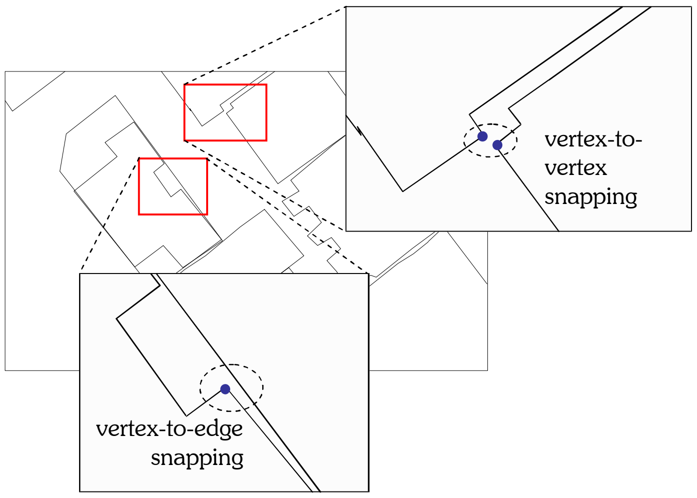


Fig. 4: Gap closing/repair.

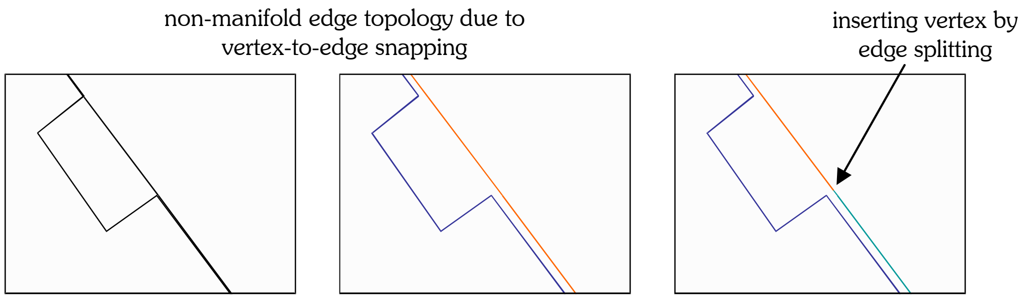

non-manifold edge topology due to vertex-to-edge snapping
inserting vertex by edge splitting

Fig. 5: Resolving non-manifold edge topology.

The result of the automatic repair process is illustrated in Fig. 6.

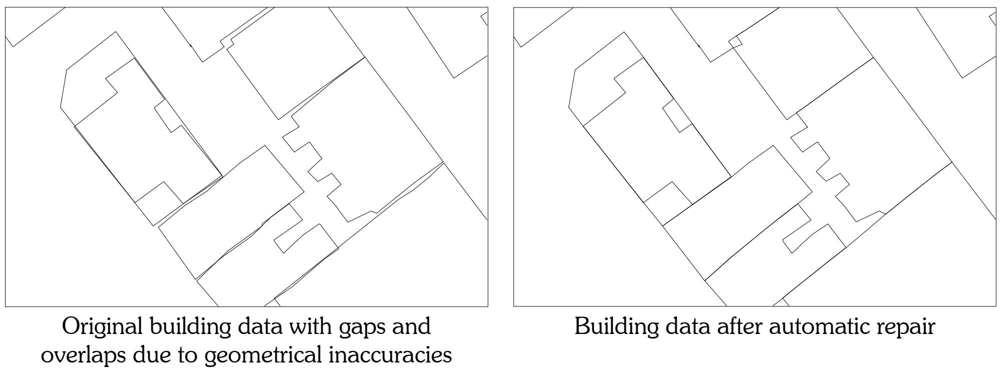

Original building data with gaps and overlaps due to geometrical inaccuracies
Building data after automatic repair

Fig. 6: Automatic repair of geometrical tolerance issues.

3. RESULTS

The algorithm described in this paper is implemented using Visual C++. An OpenGL-driven graphical interface is used to wrap the algorithm for easy user interaction. The performance benchmarking of the algorithm is conducted on a Pentium IV 3GHz machine with 1GB of RAM.

First, a comparison was made between the performance of the Douglas-Peucker algorithm and our algorithm. Fig. 7 illustrates the results of these two simplification strategies and Tab. 1 indicates the performances of the two methods. In this test, the automatic clustering option of our algorithm is switched off so that the comparison can be made fairly on a per-polygon basis. The percentage reduction in the number of edges using the Douglas-Peucker algorithm is 60.7% of the original number of edges while that using our algorithm is 68.9%. Moreover, the reduction of short edges (i.e. edges shorter than 5m) using the Douglas-Peucker algorithm is 85.2% of the original number of edges while that using our algorithm is 94.1%. Even though our method of using the image morphology takes a slightly longer time to execute, the extra time is hardly of any significance as compared to the potential reduction in the computational cost.

It is found that vector-based simplifications, such as the Douglas-Peucker algorithm, perform faster than image-based simulation and that the rasterisation from vectors to images is likely to cause a higher approximation error. However, an image-based approach will enable the ease of performing complex building clustering and merging. It can handle any complex 2D geometry easily. The approximation error can be reduced by using a finer resolution during the rasterisation process. The higher approximation error is tolerable and in fact encouraging, as it is feasible for model reduction especially when the models are used for large scale urban-buildup CFD simulation. While talking about simulating a region of several square kilometers or more, the minute details are not important. In fact, one criteria for such CFD modeling is to reduce edges and surfaces as well as fine details that hinders the creation of suitable simulation meshed models.

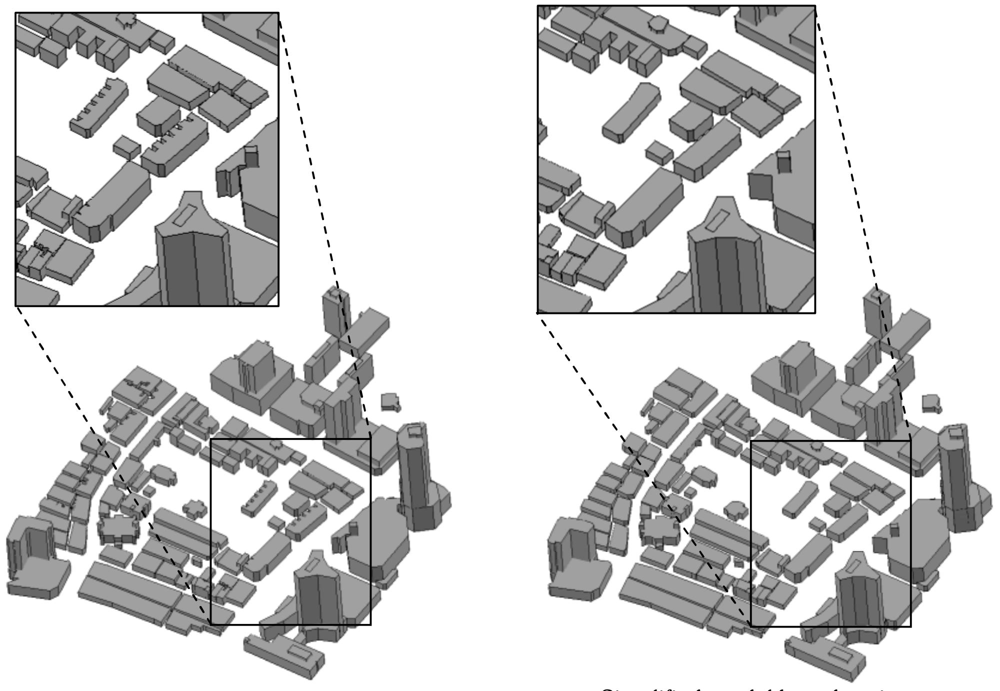

Simplified model based on Douglas-Peucker algorithm
Simplified model based on image morphology method

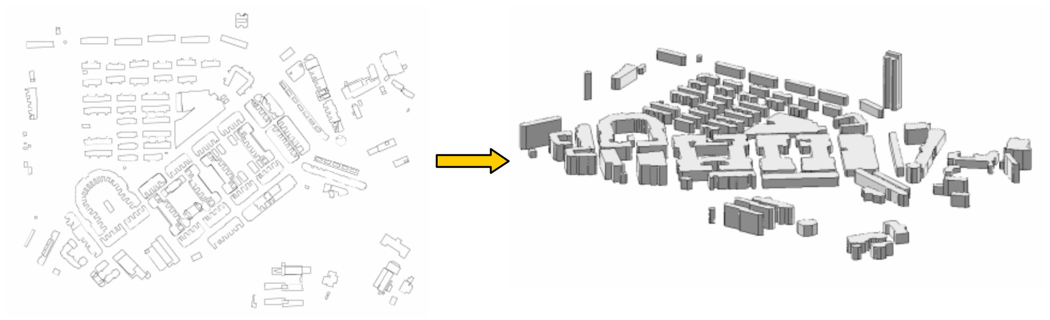
Fig. 7: Results of different simplification strategies.

	Original Model	Simplified Model (Douglas-Peucker)	Simplified Model (Image morphology)
No. of buildings	141	123	123
No. of edges	2529	993	788
No. of edges < 5 metres	1436	213	85
Exported IGES file size	2,571 kb	1,156 kb	972 kb
Time to complete	N.A.	0.047 (s)	8.7 (s)

Tab. 1: Performance comparison.

Next, we illustrate the robustness of our algorithm by applying the simplification on two other case studies which contain more buildings. The first case study (as illustrated in Fig. 8) contains 228 buildings with a total of 11,958 edges in the polygon representation. The algorithm took less than 2mins to simplify the model and the number of small edges has been drastically reduced from 2,402 to 65. This is a huge saving in terms of time as the manual method of repairing and simplification took around 2 weeks to complete. In the second case study (as illustrated in Fig. 9), the GIS data contains 887 buildings with a total of 49,002 edges in the polygon representation. The algorithm took approximately 8mins to simplify the model and the number of small edges has been reduced from 8,062 to 495. The benchmark results for the 2 case studies are presented in Tab. 2 and 3.

In all these cases, the 3D model is exported as an IGES file with each building represented as a BREP entity. The IGES data is then imported into ANSYS CFX Workbench for mesh generation.

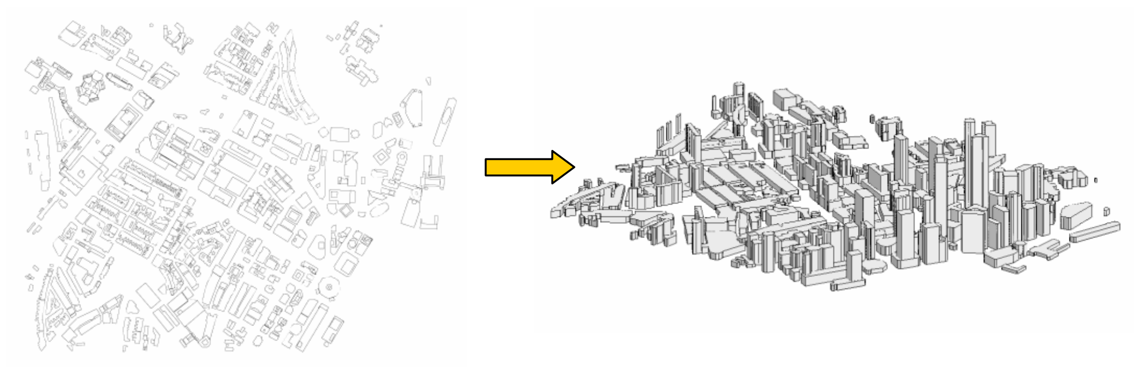


GIS data Mesh-friendly CAD model

Fig. 8: Case Study 1.

	Original Model	Simplified Model
No. of buildings	228	85
No. of edges	11958	3348
No. of edges < 5 meters	2404	69
IGES file size	4,066 kb	1,278 kb
Time to complete	N.A.	112 (s)

Tab. 2: Performance benchmarks for Case Study 1.



GIS data   Mesh-friendly CAD model

Fig. 9: Case Study 2.

	Original Model	Simplified Model
No. of buildings	887	323
No. of edges	49002	10983
No. of edges < 5 metres	8062	495
IGES file size	13,903 kb	4,599 kb
Time to complete	N.A.	494 (s)

Tab. 3: Performance benchmarks for Case Study 2.

Parameters for all case-studies are listed as follows:

i. Small building's Tolerance (area), B_a = 10 m^2
ii. Low Building's Tolerance (height), B_h= 5 m
iii. Pixel dimension (pixel per unit area), PPA = 1m × 1m
iv. Building's clustering tolerance (gap tolerance), T_c= 5 m
v. Building's feature simplification tolerance (height tolerance), T_f= 5 m
vi. Building's merging tolerance (height tolerance), T_m= 10 m
vii. Gap-repair Tolerance, T_{gap}= 1.5 m

4. CONCLUSIONS

In this work, we have gained in-depth understanding of the various issues regarding the interpretation of the ESRI file structure and format, and have successfully implemented the shapefile interpreter and IGES converter. The various algorithms to support automatic model simplification have also been implemented. These include algorithms to deal with tolerance issues, polygonal simplification and building clustering. The various algorithms were integrated into a Windows-based GUI which allows intuitive user interaction. Other basic editing function and auxiliary functions were also included in the current version of the map editor software.

5. REFERENCES

[1] Environmental Systems Research Institute, http://www.esri.com/
[2] Pitney Bowes MapInfo Corporation, http://www.mapinfo.com/
[3] Geographic Resources Analysis Support System, http://grass.itc.it/
[4] Chu, A. K. M.; Kwok, R. C. W.; Yu, K. N.: Study of pollution dispersion in urban areas using Computational Fluid Dynamics (CFD) and Geographic Information System (GIS), Environmental Modelling & Software, 20(3), 2005, 273-277.

[5] ANSYS CFX, http://www.ansys.com/products/cfx.asp

[6] Coirier, W. J.; Kim, S.: CFD Modeling for Urban Area Contaminant Transport and Dispersion: Model Description and Data Requirements, Sixth Symposium on the Urban Environment, JP2.11, 2006.

[7] Baffour, R.; Aliabadi, S.; Ji, A.: GIS based Complex 3D Geometric Modeling for High Performance Computing Chemical/Biological Dispersion Simulation, Proceedings of Environmental Modelling and Simulation, EMS 2004, 432-058.

[8] Sugihara, K.: GIS-based Automatic Generation of 3-D Building Model from Building Polygons Filtered, Proceedings of Environmental Modelling and Simulation, EMS 2004, 432-040.

[9] Baker, B.; Semwal, S. K.: 3D Visualization of 2D Topographic Data, Proceedings of Environmental Modelling and Simulation, EMS 2004, 432-073.

[10] Glander, T.; Döllner J.: Cell-based Generalization of 3D building groups with outlier management, Proceedings of the 15th International Symposium on Advances in Geographic Information Systems, Article No. 54, 2007.

[11] Anders, K. H.: Level of Detail Generation of 3D Building Groups by Aggregation and Typification, Proc. 22nd Int'l Cartographic Conf., 2005.

[12] Forberg, A.: Simplification of 3D Building Data, ICA Workshop on Generalization and Multiple representation, 2004.

[13] Forberg, A.; Mayer, H.: Generalization of 3D Building Data Based on Scale-Spaces, ISPRS Journal of Photogrammetry and Remote Sensing, 62(2), June 2007, 104-111.

[14] ESRI Shapefile Technical Description - ESRI White Paper, July 1998, http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

[15] Douglas, D.; Peucker, T.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, The Canadian Cartographer, 10(2), 1973, 112-122.

[16] Pavlidis, T.: Algorithms for Graphics and Image Processing, Computer Science Press, Rockville, Maryland, 1982.

[17] Gonzalez, R. C.; Woods, R. E.: Digital Image Processing, TA1632.G66 1992, ISBN 0-201-60078-1.