LEAST-SQUARES BUILDING MODEL FITTING USING AERIAL PHOTOS AND LIDAR DATA

Sendo Wang[1]**,** Post-doctoral Research Fellow
Yi-Hsing Tseng[2]**,** Professor
Ayman F. Habib[3]**,** Professor
[1,3]Department of Geomatics Engineering, University of Calgary
2500 University Drive N.W., Calgary, Alberta, Canada T2N 1N4
{sendo, ahabib}@ucalgary.ca

[2]Department of Geomatics, National Cheng Kung University
1 University Road, Tainan City, Taiwan, ROC 70101
tseng@mail.ncku.edu.tw

ABSTRACT

Building models are conventionally reconstructed by measuring their vertices point-by-point in a *digital photogrammetric workstation* (DPW), which is time and labor consuming process. Although aerial photos implicitly provide 3D information of buildings, LiDAR systems directly provide high density and accurate point cloud coordinates. However, LiDAR data cannot accurately represent the building boundaries. To take advantage of both systems, we propose *Floating Model* and a tailored *least-squares model-data fitting* (LSMDF) algorithm in this paper. The floating model is a pre-defined primitive model, which is described by a set of parameters, floating in the space. A building is reconstructed by adjusting these model parameters so the wire-frame model adequately fits the building's boundary in all overlapping photos and LiDAR data. The semi-automated modeling procedure consists of 3 steps. First, the operator chooses an appropriate model and approximately fit it to the building's outlines on the aerial photos. Then, an automated procedure computes the optimal fit between the models and both of aerial photos and LiDAR data using an iterative LSMDF algorithm. Finally, the model parameters and standard deviations are provided, and the wire-frame model is superimposed on all overlapping aerial photos for the operator to check or modify the results. To test the proposed algorithm and approach, an image block of 4 panchromatic aerial photos and corresponding LiDAR data are selected for the experiments. The ground resolution of the image is approximately 5*cm*. The point density of LiDAR point cloud is about 4-5*point*/*m*^{2}. The reconstructed models are manually evaluated and compared. Most of the buildings are accurately modeled, and the fitting result achieves the photogrammetric accuracy. In addition, the implicit constraints within the model, such as the parallel edges or rectangle corners, will keep the building shape without distortion.

INTRODUCTION

Digital Building Models (DBM) are the most essential information for many modern applications, such as urban planning and management (Steinicke *et al.*, 2006), mobile navigation (Rakkolainen and Vainio, 2001), wireless telecommunication (Wahl *et al.*, 2005), tourism promotion (Berlin, 2010), and true orthophoto generation (Habib *et al.*, 2007) etc. Efficient acquisition of 3D city objects has become a more and more popular topic (Braun *et al.*, 1995; Brenner, 2005; Englert and Gülch, 1996; Grün, 2000; Kim and Habib, 2009; Lang and Förstner, 1996). Conventional photogrammetry concentrates on the accurate reconstruction of 3D coordinates of points. Current automated systems set up by image matching algorithms are mainly based on the point-to-point correspondence. However, higher-order features such as linear, planar or volumetric primitives contain more geometric and semantic information than a single point. That draws the research community's intention to use 3D model as a modeling tool for extracting objects from source data.

Although the CAD system is not initially developed for photogrammetric purpose, its powerful functions of drawing, manipulating, and visualizing 2D objects have made it widely used in photogrammetric systems. The increasing demands of object's 3D models inspires many research toward using 3D CAD models as a modeling tool to reconstruct objects from image data (Bhanu *et al.*, 1997; Böhm *et al.*, 2000; Brenner, 2000; Das *et al.*, 1996, Ermes *et al.*, 1999, Tseng and Wang, 2003, van den Heuvel, 2000, Vosselman, 1999). This trend towards integration of photogrammetry and CAD system in the algorithmic aspect creates a new term: "*CAD-based Photogrammetry*". Previous research has shown that using CAD models does increase the efficiency of photogrammetric modeling both by the advanced object modeling techniques, such as *Constructive Solid Geometry* (CSG), and the incorporation of geometric object constraints.

Building modeling usually involves high-level information (model knowledge) and low-level data (images or LiDAR point cloud). Processing low-level data to obtain higher-level information is a *bottom-up* or *data-driven* procedure, and reversely a *top-down* or *model-driven* procedure derives features from high-level information and verifies the correspondence between the derived features and low-level data. Both bottom-up and top-down procedures are more or less applied in a building modeling approach. Information and data meet in a certain level for the verification of correspondence. In general, a fully-automated building modeling approach tends to verify the correspondence in the higher level than a semi-automated approach. A hypothesis test or engineering knowledge procedure is required for full-automated methods to determine the most appropriate model with respect to the scene (Baillard and Zisserman, 1999; Henricsson *et al*., 1996; Kim and Habib, 2009; Lu *et al*., 2006; Suveg and Vosselman, 2004). *Model-based building reconstruction* (MBBR) is rather a top-down approach which starts with hypotheses of building model representing a specified target on the scene, and verifies the compatibility between the model and the available data sources, such as topographic maps, aerial photos, LiDAR data, and DEM (Ameri, 2000; Brenner, 1999; Sester and Förstner, 1989; Wang and Tseng, 2004). Most of the MBBR approaches are implemented in a semi-automated manner, solving the model-data fitting problem based on some high-level information given by the operator, for examples, inJECT (Förstner 1999), CC-Modeler (Grün *et al*., 2002), ATOP (Brenner, 2005), and PhotoModeler (EOS Systems, 2010). Semi-automated approaches leave the high level tasks, such as model hypothesis, model selection, and model detection, to humans to ensure the quality of data interpretation, while performing model-image fitting automatically to improve the modeling efficiency. This cooperation would make semi-automated building reconstruction practically valuable.

Inspired by CAD-based photogrammetry and MBBR, we propose a complete 3D building modeling approach – *Floating Models* – for reconstructing building models simultaneously from aerial photos (2D) and LiDAR point cloud (3D). The floating model represents a flexible entity floating in 3D space. It can be a point, a line segment, a surface plane, or a volumetric model. Each model is associated with a set of shape parameters and a set of pose parameters. The shape parameters describe the model's volume in the designated dimension. The pose parameters define the datum point's position and the orientation of the model. From the traditional photogrammetric point of view, the floating models are extensions of the floating mark. However, the floating model does not only float in the object space indicating a position, but also can be adjusted in terms of size and orientation to represent the 3D outlines of an object.

MBBR relies on a model-data fitting algorithm to obtain the optimal fit between model and data sources. Attempts to solve the problem of model-image fitting date back to the work of Sester and Förstner (1989). By fitting the projected model onto the image, the transformation parameters of the building model are determined using a clustering algorithm followed by a robust estimation of model parameters. This budding research has marked an important step toward MBBR, although the algorithm is restricted to fit a model onto one single image rather than multiple images. Concurrently in the field of computer vision for model-based vision, Lowe (1987) proposed the *least-squares model-image fitting* (LSMIF) algorithm to determine the projection and model parameters that best fit a 3D-model to matching 2D-image features. Lowe's study developed the fundamental theory of the LSMIF for generic applications. This rigorous fitting algorithm has been recognized as a key to dealing with MBBR (Veldhuis, 1998; Vosselman and Veldhuis, 1999).

The previous work (Tseng and Wang, 2003; Wang and Tseng, 2004; Wang and Tseng, 2009) has successfully applied the CSG principle and a tailored least-squares model-image fitting algorithm to modeling versatile buildings merely from aerial photos. Buildings are reconstructed semi-automatically by adjusting the model parameters to fit the model to all overlapped images. In this paper, we improve the LSMIF as *least-squares model-data fitting* (LSMDF) algorithm in order to incorporate other data sources. The model parameters are determined by fitting the model both to the aerial photos and to the LiDAR point cloud. To simplify the fitting problem, the model parameters are rearranged into two groups, horizontal and vertical parameters. The proposed semi-automated reconstruction procedures are also divided into three steps. First, the operator chooses an appropriate model according to the aerial photos and approximately fit it to the building's outlines on the aerial photos. Second, the LSMDF algorithm iteratively computes the horizontal parameters by fitting the projected wireframe model to the extracted edge pixels on aerial photos, and computes the vertical parameters by fitting the model's roof patches to the extracted LiDAR point cloud. The horizontal and vertical parameters are computed in sequence iteratively to approach the optimal fit. Finally, the wireframe model is re-projected onto aerial photos for verification. The operator can make further modification according to the photos if necessary. Figure 1 uses a box model as an example to depict the proposed semi-automated modeling procedures. The hexagons depict the spatial information provided from the data sources.

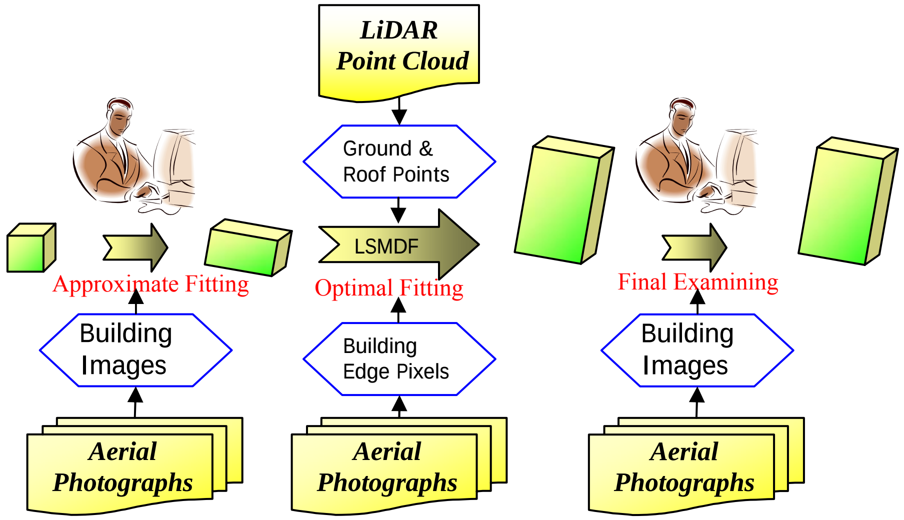
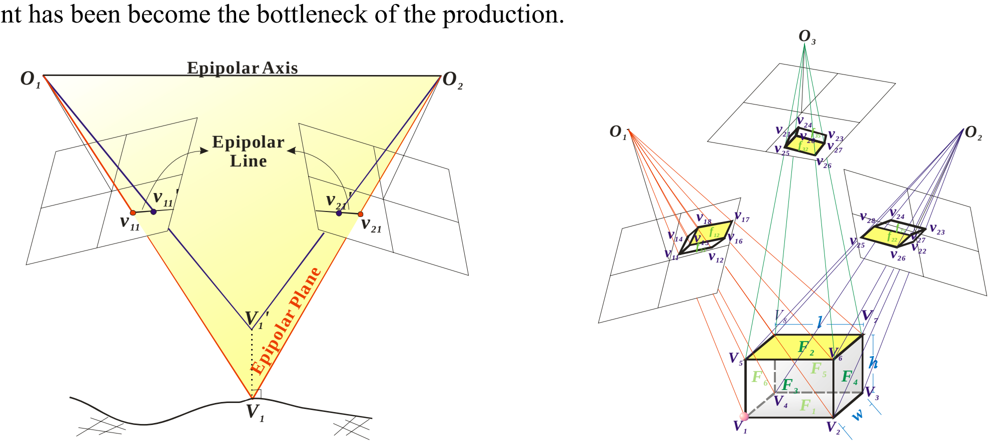


Figure 1. The proposed modeling procedures start from manually fitting model to image approximately. Then, the optimal fit among aerial photos and LiDAR point cloud is calculated by LSMDF algorithms. Finally, the model is re-projected onto aerial photos for verification.

FLOATING MODELS

Conventional photogrammetric mapping systems concentrate on the accurate reconstruction of 3D points. The floating mark is a simple way to represent the position of a point in the space, and thus, has been utilized as the only tool in the stereo plotters up to now. The idea behind the floating mark is to recover the intersection ground point V_1 of the bundles from the projection centers O_1 and O_2, through the image point v_{11} and v_{21}, as figure 2(a) shows. If the conjugate point v_{11}' or v_{21}' moves along the conjugate epipolar lines, the intersection point V_1 represented by floating mark will rise or sink along the epipolar plane, seems like "floating" in the object space. This simple representation of a 3D coordinates has been very useful for photogrammetric measurement and 2.5D mapping system. However, the floating mark reaches its limits when the conjugate points cannot be identified due to occlusions or interferences from other objects. With the increasing needs for 3D object models, point-by-point measurement has been become the bottleneck of the production.



(a) Floating Mark

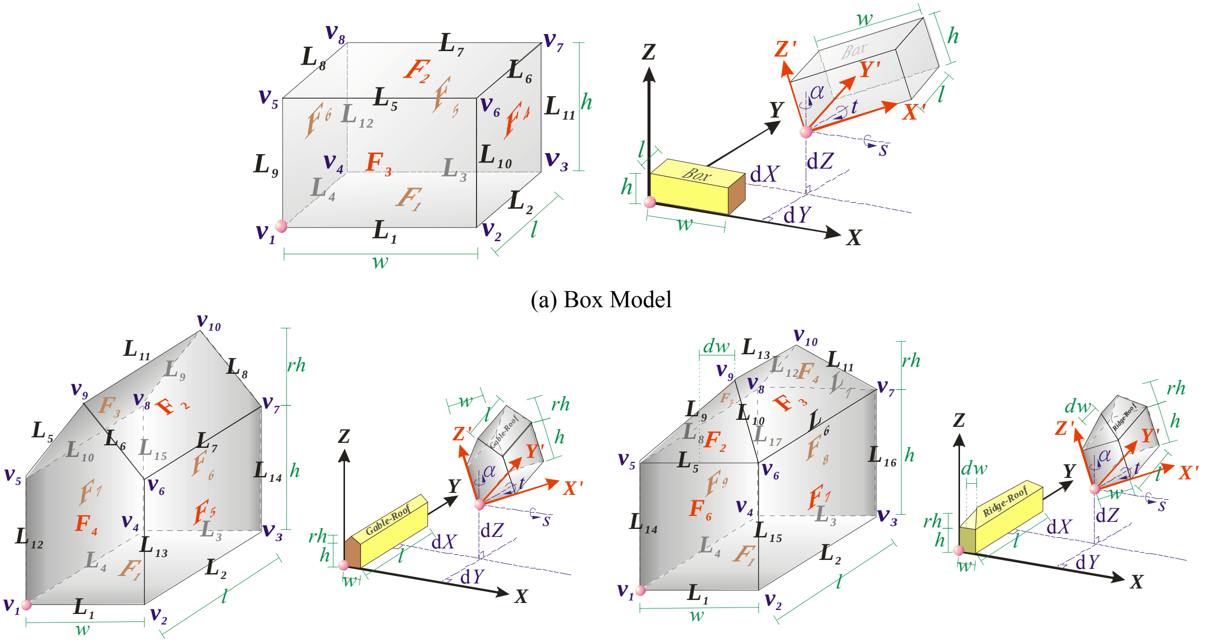
(b) Floating Model

Figure 2. The geometry of floating mark and floating model.

To deal with the modeling problem, we propose the floating model as the 3D measuring tool that complies with the *constructive solid geometry* (CSG). Each floating model is basically a primitive model, which determines the intrinsic geometric property of a part of the building. The primitive model could be any kind of practical models as long as it can be defined and represented by parameters. For example, it could be line segment, rectangle plane, circular plane, triangular plane, box, or gable-roof house etc. Take the box primitive in figure 2(b) as an example, the floating model does not only recover one intersection V_1 from one set of bundles, but also many intersections (V_1, V_2, V_3, …) distributed on every edge and every face of the model from multiple sets of bundles. These redundant bundles result in much better reliability. More importantly, the model's inner geometric characteristics, such as parallel edges, are implicitly considered by recovering all intersections simultaneously. Therefore, it is capable to reconstruct a building which is partially occluded.

Despite the variety in their shape, each primitive model commonly has a datum point, and is associated with a set of pose parameters and a set of shape parameters. The datum point and the pose parameters determine the position and pose of the floating model in object space. It is adequate to use 3 translation parameters (dX, dY, dZ) to represent the datum point's position and 3 rotation parameters, swing (s) around X-axis, tilt (t) around Y-axis, and azimuth (α) around Z-axis to represent the rotation of a primitive model. The shape parameters describe the shape and size of the primitive model, *e.g.*, a box has three shape parameters: width (w), length (l), and height (h). Changing the values of shape parameters changes the dimensions of the primitive in the three designated directions, but still keeps its shape as a rectangular box. Various primitive may be associated with different shape parameters, *e.g.*, a gable-roof house primitive has an additional shape parameter – roof's height (rh).

Although models are described by parameters, they are also composed of vertices, edges, and faces. The vertex-edge-face topology is used not only for displaying models but also model fitting. Figure 3 shows the topology and the model parameters of (a) box model, (b) gable-roof model, and (c) ridge-roof model. The X'-Y'-Z' coordinate system is defined in the model space while the X-Y-Z coordinate system is defined in the object space. The little pink sphere indicates the datum point of the model. The yellow primitive model is in the original position and pose, while the grey model depicts the position and pose after adjusting parameters. It is very clear that, the model is "floating" in the space by controlling these pose parameters, and the volume size is flexible along the designated dimensions by controlling the shape parameters.



(a) Box Model

(b) Gable-roof Model

(c) Ridge-roof Model

Figure 3. Topology and the model parameters of 3 floating models.

The primitive model defined in the model space is a simple unit solid. For example, a box is a unit cube; its width, length, and height are all equal to 1. The shape parameters are used to adjust the box to the correct size, and the pose parameters are used to rotate and move the box to the correct pose and position in the object space. Since normal buildings rarely rotate around the X-axis or the Y-axis, the swing angle (s) and the tilt angle (t) are neglected for the building reconstruction purpose. Table 1 lists the coordinates of a box's 8 vertices after going through the transformation from model space to object space. Each vertex in the object space is then projected onto the aerial photos by the collinearity condition equations on the basis of the known interior and exterior orientation parameters.

Table 1. Vertices coordinates from model space to object space

Vertex No.	Model Space Coordinate	After Shape Parameters	After Rotation	After Translation (Object Space Coordinate)
v_1	(0, 0, 0)	(0, 0, 0)	(0, 0, 0)	$(\mathrm{d}X, \mathrm{d}Y, \mathrm{d}Z)$
v_2	(1, 0, 0)	$(w, 0, 0)$	$(w\cos\alpha, w\sin\alpha, 0)$	$(w\cos\alpha+\mathrm{d}X, w\sin\alpha+\mathrm{d}Y, \mathrm{d}Z)$
v_3	(1, 1, 0)	$(w, l, 0)$	$(w\cos\alpha-l\sin\alpha, w\sin\alpha+l\cos\alpha, 0)$	$(w\cos\alpha-l\sin\alpha+\mathrm{d}X, w\sin\alpha+l\cos\alpha+\mathrm{d}Y, \mathrm{d}Z)$
v_4	(0, 1, 0)	$(0, l, 0)$	$(-l\sin\alpha, l\cos\alpha, 0)$	$(-l\sin\alpha+\mathrm{d}X, l\cos\alpha+\mathrm{d}Y, \mathrm{d}Z)$
v_5	(0, 0, 1)	$(0, 0, h)$	$(0, 0, h)$	$(\mathrm{d}X, \mathrm{d}Y, h+\mathrm{d}Z)$
v_6	(1, 0, 1)	$(w, 0, h)$	$(w\cos\alpha, w\sin\alpha, h)$	$(w\cos\alpha+\mathrm{d}X, w\sin\alpha+\mathrm{d}Y, h+\mathrm{d}Z)$
v_7	(1, 1, 1)	(w, l, h)	$(w\cos\alpha-l\sin\alpha, w\sin\alpha+l\cos\alpha, h)$	$(w\cos\alpha-l\sin\alpha+\mathrm{d}X, w\sin\alpha+l\cos\alpha+\mathrm{d}Y, h+\mathrm{d}Z)$
v_8	(0, 1, 1)	$(0, l, h)$	$(-l\sin\alpha, l\cos\alpha, h)$	$(-l\sin\alpha+\mathrm{d}X, l\cos\alpha+\mathrm{d}Y, h+\mathrm{d}Z)$

LEAST-SQUARES MODEL-DATA FITTING

Based on the previous research, buildings can be successfully modeled solely from aerial photos. However, the LiDAR point cloud directly provides 3D coordinates of massive ground points with better vertical accuracy. If the LiDAR point cloud can be registered to the same datum of aerial photos, these 3D points should be able to improve the accuracy and reliability while reconstructing the building's roof and the ground height. Therefore, we expand the original LSMIF to LSMDF to incorporate LiDAR point cloud. The model parameters are rearranged into two groups: (1) horizontal parameters, such as width (w), length (l), planar position of datum point ($\mathrm{d}X$, $\mathrm{d}Y$), azimuth (α); (2) vertical parameters, such as height (h), height of datum point ($\mathrm{d}Z$), and roof height (rh) for a gable-roof model.

Assume that the orientation of aerial photos has been determined by other means and the LiDAR point cloud has been registered to the same datum of aerial photos, the visible vertices and edges of a floating model can be projected onto all available photos. The visibility of each edge on the designated photo is determined by calculating the normal vector of the face encompassing that edge. The face is visible on the photo as long as its normal vector is positive. Giving a set of model parameters, all vertices' object coordinates can be calculated and the wireframe model can be reconstructed in the LiDAR point cloud. Since the model has been approximately fit by human operator, the searching range can be defined and the initial model parameters are available for the iterative fitting. The wireframe model is first projected onto all photos to solve the horizontal parameters by fitting the projection of the wireframe model to the extracted edge pixels. Then the floating model is projected into the LiDAR point cloud to solve the vertical parameters by fitting roof patches to the point cloud.

Aerial Photo Fitting

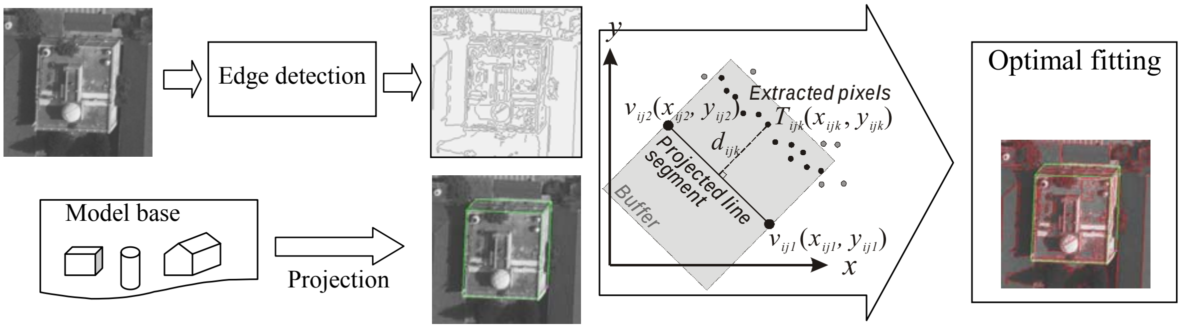


Figure 4. Setting up a buffer along every edge of the model to identify candidate edge pixels on all photos.

Although all shape and pose parameters will affect the model projection on the aerial photo, we leave the vertical parameters to be determined by fitting the model to LiDAR point cloud. At this stage, the objective of the fitting is the building's outlines appearing on the aerial photos. To obtain the first approximate model as close to the optimal fitting as possible, we develop an interface program allowing the operator to resize, rotate, and move a model to approximately fit all corresponding photos (Wang and Tseng, 2009). Using the approximate fitting, the LSMDF iteratively pulls the model to the optimal fit instead of blindly searching whole photo for the solution. To avoid the disturbance of irrelevant edge pixels, only those edge pixels distributed within the specified buffer zones are used in the calculation of the fitting algorithm. Figure 4 depicts the extracted edge pixels T_{ijk} and the buffer determined by a projected edge $\overline{v_{ij1}v_{ij2}}$ of the model. The suffix i represents the index of projected line segment from 1 to I, j represents the index of overlapped images from 1 to J, and k represents the index of the edge pixels from 1 to K. Ignoring edge pixels outside the buffer is reasonable, since the discrepancies between the projected edges and the corresponding edge pixels should be small, as the model parameters are approximately known. However, the size of the buffer has to be carefully chosen because it will directly affect the convergence of the computation, *i.e.*, the pull-in range. We propose a decreasing buffer-size strategy to improve the pull-in range of LSMDF – "The initial buffer should be larger to include more edge pixels at earlier iterations, and reduced to ensure the convergence as the iteration proceeds." According to the previous research, the initial buffer size is set to 0.5*mm* and is decreased 0.5*mm* after each iteration until it reaches the final size of 0.05*mm*.

Since the detected edge pixels are the target on photos of the LSMDF algorithm, the quality of detection is crucial to the result of the fitting. A distinct edge pixel usually has a higher gradient intensity, which can be detected by most of the edge detectors, such as the Canny edge detector. However, an indistinct edge pixel may not be detected or may be detected together with other non-edge pixels. If all extracted pixels within the buffer are equally treated as the fitting targets, the non-edge pixels could lead the fitting to a wrong position. Therefore, the detected edge pixels within the buffer should be filtered based on their gradient vector which refers to the direction of the maximum gray value difference. The ideal projection of the wire-frame model should exactly overlap on the detected edge pixels. The ideal angle between the projected line segment and the gradient vector should be 90° or 270°. A threshold of ±15° for the angle difference to 90° or to 270° is selected for the filtering purpose.

The optimal fitting condition we are looking for is that all projected edges coincide with the building boundaries shown on aerial photos. In Equation (1), the distance d_{ijk} represents a discrepancy between a sample point T_{ijk} and its corresponding edge $\overline{v_{ij1}v_{ij2}}$, which is expected to be zero. Therefore, the objective of the fitting function is to minimize the squares sum of d_{ijk}. Suppose an edge is composed of the vertices $v_{ij1}(x_{ij1}, y_{ij1})$ and $v_{ij2}(x_{ij2}, y_{ij2})$, and there is an edge pixel $T_{ijk}(x_{ijk}, y_{ijk})$ located inside the buffer. The distance d_{ijk} from the point T_{ijk} to the edge $\overline{v_{ij1}v_{ij2}}$ can be formulated as the following equation:

$$d_{ijk} = \frac{\left|(y_{ij1} - y_{ij2})x_{jk} + (x_{ij2} - x_{ij1})y_{jk} + (y_{ij2}x_{ij1} - y_{ij1}x_{ij2})\right|}{\sqrt{(x_{ij1} - x_{ij2})^2 + (y_{ij1} - y_{ij2})^2}} \tag{1}$$

The coordinates of vertices $v_{ij1}(x_{ij1}, y_{ij1})$ and $v_{ij2}(x_{ij2}, y_{ij2})$ are actually functions of the unknown parameters, as table 1 shows. Therefore, d_{ijk} will be a function of the shape and pose parameters of the selected model. Taking a box model for instance, d_{ijk} will be a function of w, l, h, α, dX, dY, and dZ, with the hypothesis that a normal building rarely has a tilt (t) or a swing (s) rotation. The least-squares solution for the unknown parameters can be expressed as:

$$\Sigma d_{ijk}^2 = \Sigma[F_{ijk}\ (\ w, l, h, \alpha, \mathrm{d}X, \mathrm{d}Y, \mathrm{d}Z)]^2 \ \rightarrow min. \tag{2}$$

Equation (2) is a nonlinear function with regard to the unknowns, therefore the Newton's progressive method and the 1st order Taylor's series expansion are applied to solve for the unknowns. The nonlinear function is differentiated with respect to the unknowns and becomes a linear function by ignoring the second and higher order terms with regard to the increments of unknown parameters as the following equation demonstrates:

$$d_{ijk} - F_{ijk0} = \left(\frac{\partial F_{ijk}}{\partial w}\right)_0 \Delta w + \left(\frac{\partial F_{ijk}}{\partial l}\right)_0 \Delta l + \left(\frac{\partial F_{ijk}}{\partial h}\right)_0 \Delta h + \left(\frac{\partial F_{ijk}}{\partial \alpha}\right)_0 \Delta \alpha + \left(\frac{\partial F_{ijk}}{\partial dX}\right)_0 \Delta dX + \left(\frac{\partial F_{ijk}}{\partial dY}\right)_0 \Delta dY + \left(\frac{\partial F_{ijk}}{\partial dZ}\right)_0 \Delta dZ \tag{3}$$

where F_{ijk0} is the approximation of the function F_{ijk} calculated with the given approximations of the model parameters. $\left(\frac{\partial F_{ijk}}{\partial w}\right)_0, \left(\frac{\partial F_{ijk}}{\partial l}\right)_0, \left(\frac{\partial F_{ijk}}{\partial h}\right)_0, \ldots$ are partial derivatives of equation 1 with respect to unknown parameters and are calculated with the approximated model parameters. Δw, Δl, Δh, $\Delta \alpha$, ΔdX, ΔdY, ΔdZ are the unknowns in equation 3 referring to the increments of the model parameters. Since the focus of photo fitting is on solving the horizontal parameters, we suppose all increments of vertical parameters, such as Δh and ΔdZ, are zero. Thus, the practical observation function of photo fitting becomes:

$$d_{ijk} - F_{ijk0} = \left(\frac{\partial F_{ijk}}{\partial w}\right)_0 \Delta w + \left(\frac{\partial F_{ijk}}{\partial l}\right)_0 \Delta l + \left(\frac{\partial F_{ijk}}{\partial \alpha}\right)_0 \Delta \alpha + \left(\frac{\partial F_{ijk}}{\partial dX}\right)_0 \Delta dX + \left(\frac{\partial F_{ijk}}{\partial dY}\right)_0 \Delta dY \tag{4}$$

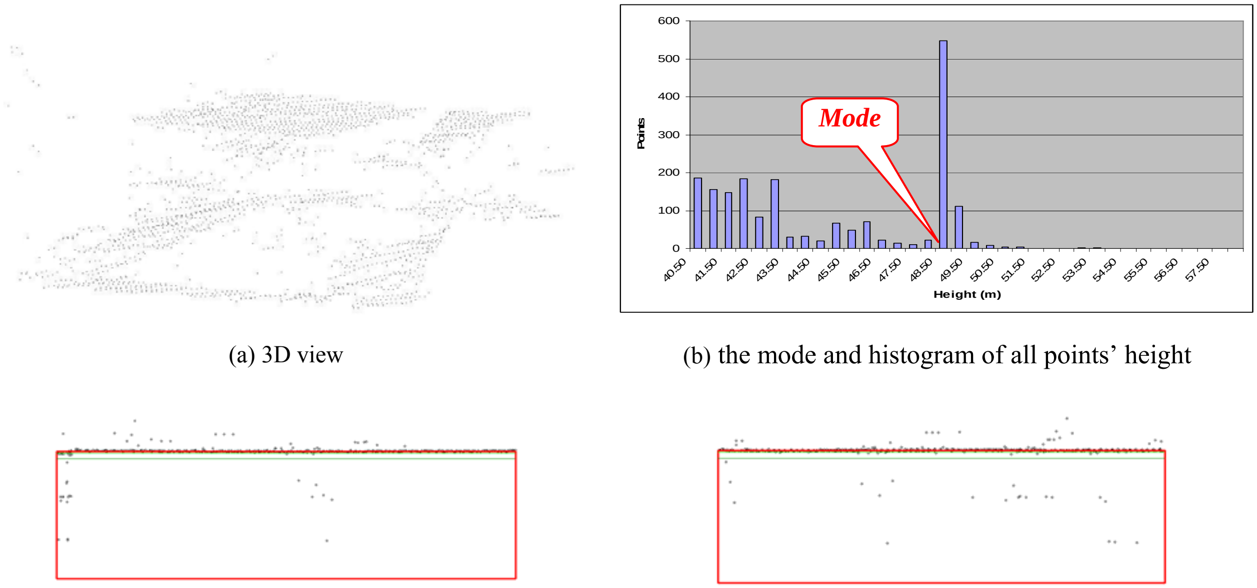
Given a set of approximates for unknowns, the least-squares solution for the unknown increments can be calculated, and the approximations are updated by the increments. Each edge pixel within the buffer provides an observation function as equation 4. All observations are considered in the least-squares adjustment, which means there would be $I*J*K$ observations. Although some edge pixels may be included twice by two adjacent projected line segments, the iteration will lead to the optimal solution for the whole model.

After fitting to the LiDAR point cloud for solving the vertical parameters, repeating this calculation to update the unknown horizontal parameters. The horizontal and vertical parameters are updated in sequence iteratively. The linearized equations can also be expressed as a matrix form: $\boldsymbol{V}=\boldsymbol{AX}-\boldsymbol{L}$, where $\boldsymbol{A}$ is the matrix of partial derivatives; $\boldsymbol{X}$ is the vector of the increments; $\boldsymbol{L}$ is the vector of approximations; and $\boldsymbol{V}$ is the vector of residuals. The objective function actually can be expressed as $q=\boldsymbol{V}^T\boldsymbol{V}\rightarrow min$. After each iteration, $\boldsymbol{X}$ can be solved by the matrix operation: $\boldsymbol{X}=(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{L}$. The standard deviation of each increment can also be calculated as an accuracy index of the LSMDF.

LiDAR Point Cloud Fitting

Most of the relevant research adopts 3D plane fitting algorithms to determine the roof patches of the model. In this paper, we propose a coordinate transformation approach to simplify the fitting problem from 3D to 2D. Since the horizontal parameters have been determined by fitting to aerial photos, the location of the datum point and the horizontal range of the building are determined. The height (dZ) of the datum point could be estimated by the lowest point around the building. Then, the building height (h) and the roof's height (rh) are determined by fitting the model to LiDAR point cloud extracted for the building.

For the flat roof model, such as the rectangular box model, the rooftop's height (h + dZ) is estimated by calculating the mode among all of the extracted LiDAR point's height, as Figure 5 shows. The dark green line shows the mode and the lime green line shows the average in the profile views. For the gable-roof model or the ridge-roof model, the extracted LiDAR point cloud is transformed to a local X'-Y'-Z' coordinate system defined on the lateral side of the building. Under the ideal condition, the LiDAR points should fall on the roof and form two roof eaves in the X'-Z' profile view, as Figure 6 depicts. With the coordinate transformation, the observation function of the vertical fitting is simplified as the distance from 2D point to edge, similar to the function of the horizontal fitting.



(a) 3D view

(b) the mode and histogram of all points' height

(c) X'-Z' profile view

(d) Y'-Z' profile view

Figure 5. Fitting a box model's rooftop in a local X'-Z' or Y'-Z' coordinate system by calculating the mode among LiDAR point cloud's heights.

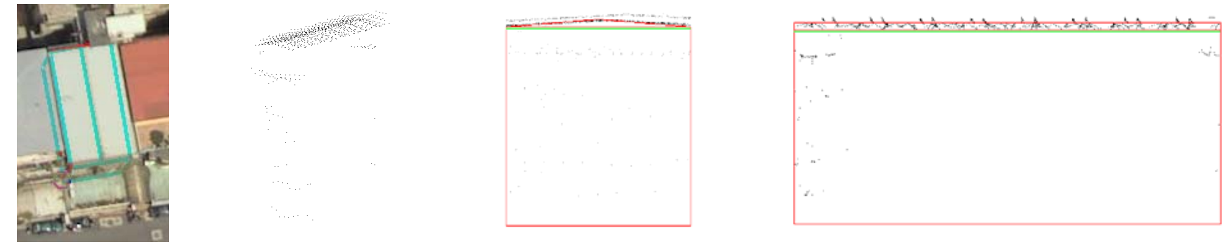

(a) Projected model on photo (b) 3D view of LiDAR point cloud (c) *X'-Z'* profile view of LiDAR point cloud (d) *Y'-Z'* profile view of LiDAR point cloud

Figure 6. Fitting roof eaves to the LiDAR point cloud in a local *X'-Z'* coordinate system.

EXPERIMENTS

To verify the capability of proposed LSMDF algorithm, several experiments have been implemented. A building is modeled by using solely aerial photos and by both aerial photos and LiDAR data. The progress of the iterations is provided to study the impact of several factors, such as buffer size, constrained a parameter, and complements from versatile data sources.

Data Sources and Program Introduction

The test data includes 4 panchromatic aerial photos and the overlapping airborne LiDAR point cloud, provided by the EuroSDR "Registration Quality – Towards Integration of Laser Scanning and Photogrammetry" project. Figure 7 shows the overview of LiDAR point cloud superimposed on the 4 aerial photos. The original photos were taken with a 120.00*mm*-focal-length DMC digital aerial camera in September 2005. The image format is 13824*pixel* * 7680*pixel* and the ground resolution is about 5*cm*. The original pixel depth is 16*bit* and has been down-resampling to 8*bit* before the experiments. The end-lap between photos is about 60%, and the side-lap is about 20%. The inner and exterior orientation information is provided from previous aerial triangulation results by the data provider. The LiDAR point cloud is collected by Leica ALS50-II airborne laser scanner in April 2007. The flying height is approximately 500*m* which results in the point density of 4-5$point/m^2$. However, the original orientation was altered for the registration purpose. The point cloud used for this paper has been registered to the same datum of the aerial photos according to the proposed methodology by Habib *et al.* (2009).

A PC program developed by C++ language is designed to implement the proposed least-squares building modeling procedures. It provides an interface to display aerial photos and to interact with the operator for all model operations, as illustrated by figure 8. A proposed modeling procedure is as follows: (1) the operator selects an appropriate model from the model base. (2) The program automatically projects the initial model onto all photos in the middle of the viewport. (3) The operator adjusts the model parameters to approximately fit model to all photos. (4) The operator clicks the "Optimal Fitting" button and the program calculates the optimal fit both to aerial photos and LiDAR point cloud iteratively. (5) The optimal fit wireframe model is projected on all photos and the final model parameters are shown on the panel and the relevant statistics data, such as iteration number and the variance of the parameter, is listed in the result window. So the operator can verify or modify the final results. In such a semi-automated manner, a building model is usually reconstructed within a minute, but the time for the whole building depends on its complexity.

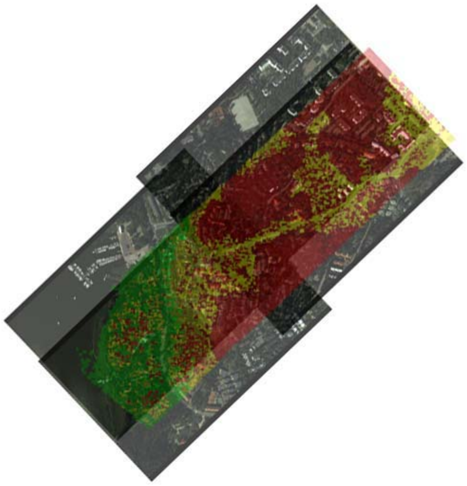
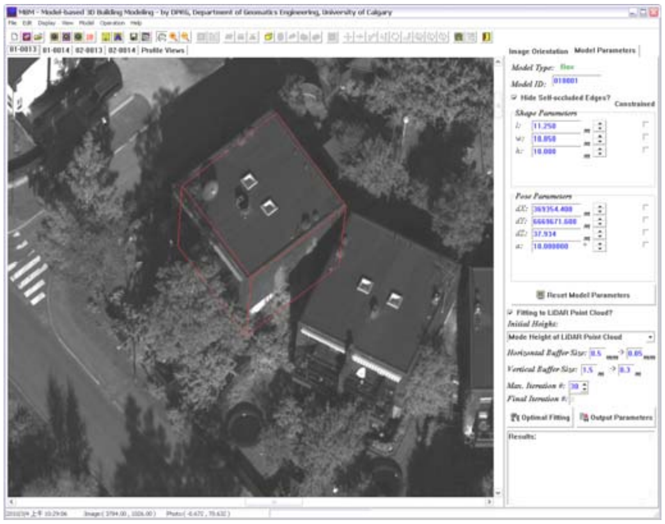


Figure 7. Test data: the airborne LiDAR point cloud superimposed on the 4 aerial photos.



MBM - Model-based 3D Building Modelling - by DPRG, Department of Geomatics Engineering, University of Calgary

File Edit Display View Model Operation Help
01-0013 01-0014 02-0013 02-0014 Profile Views
Image Orientation Model Parameters
Reset Model Parameters
Optimal Fitting
Output Parameters

Model Type: Box
Model ID: 010001
☑ Hide Self-occluded Edges?
Constrained
Shape Parameters
l: 11.250 m
w: 10.050 m
h: 10.000 m
Pose Parameters
dX: 369354.400 m
dY: 6669671.600 m
dZ: 37.934 m
α: 10.000000 °
☑ Fitting to LiDAR Point Cloud?
Initial Height: Mode Height of LiDAR Point Cloud
Horizontal Buffer Size: 0.5 mm → 0.05 mm
Vertical Buffer Size: 1.5 m → 0.3 m
Max. Iteration #: 30
Final Iteration #:
Results:

Figure 8. The program interface of the proposed least-squares building modeling approach.

Modeling Solely by Aerial Photos

Take the rectangular box-like building in figure 9 for example; this building is visible on 3 aerial photos – 01-013, 01-014, 02-013. A box primitive model is selected and manually fit on photos approximately, as the superimposed red wire-frame model shown. The initial horizontal buffer on photos is set to 0.5*mm* and will decrease 0.05mm after each iteration, until it reaches the minimum value of 0.05*mm*. The edge pixels are extracted by *Sobel Operator* with a filtering threshold of 30, and are depicted by green pixels on photos.

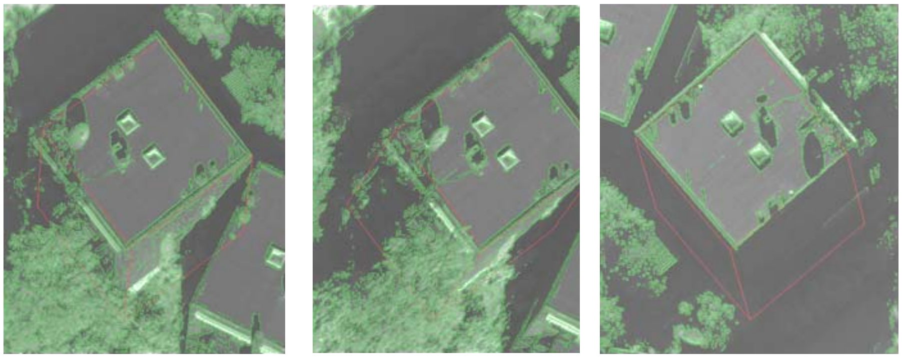


Photo 01-0013 Photo 01-0014 Photo 02-0013

l = 11.582 *m*
w = 10.347*m*
h = 8.179 *m*
d*X* = 369354.285 *m*
d*Y* = 6669671.610 *m*
d*Z* = 40.290 *m*
α = 8.473848°

Edge Pixel No. within Buffer:
Photo 01-0013: 33256
Photo 01-0014: 30254
Photo 02-0013: 18918

Figure 9. The approximate fit model and the extracted edge pixels superimposed on aerial photos.

The convergence criteria for the horizontal parameters is their increment must be smaller than 0.1m, for the vertical parameters is their increment must be smaller than 0.2m, and for the rotation angle is its increment must be smaller than 0.0001˚. To prevent from divergent, we also set the maximum iteration number as 50. After 30 iterations, the model achieves optimal fit. Figure 10 shows the optimal fit results. The number of extracted edge pixels within buffers decreases with the step-by-step decreasing horizontal buffer size. It does only prevent the model from pulling by non-edge pixels but also improve the convergence of the least-squares fitting.

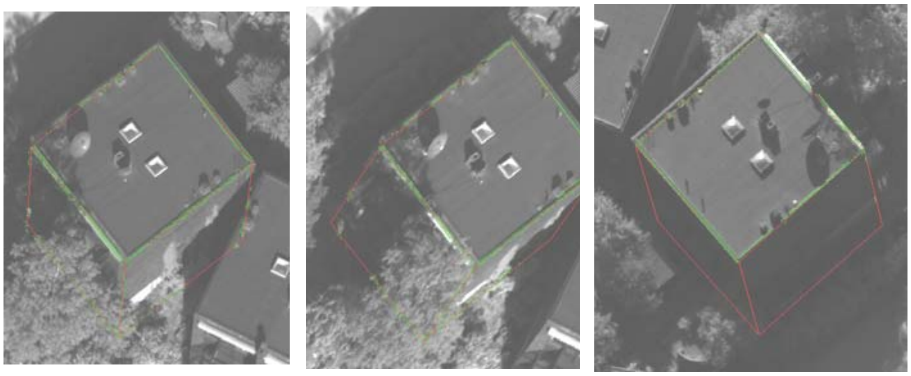


Photo 01-0013 Photo 01-0014 Photo 02-0013

$l = 11.807m$	$\sigma^2_{\Delta l} = 0.0245$
$w = 10.838m$	$\sigma^2_{\Delta w} = 0.0282$
$h = 10.017m$	$\sigma^2_{\Delta h} = 0.0001$
$\mathrm{d}X = 369353.679m$	$\sigma^2_{\Delta \mathrm{d}X} = 0.0130$
$\mathrm{d}Y=6669671.715m$	$\sigma^2_{\Delta \mathrm{d}Y} = 0.0161$
$\mathrm{d}Z = 37.959m$	$\sigma^2_{\Delta \mathrm{d}Z} = 0.0001$
$\alpha = 8.519016^\circ$	$\sigma^2_{\Delta \alpha} = 0.0002$

Edge Pixel No. within Buffer:
Photo 01-0013: 2332
Photo 01-0014: 2093
Photo 02-0013: 1417

Figure 10. The box primitive model achieves optimal fit after 30 iterations.

Constrained Parameters

The rectangular box-like building in figure 9 is also an example showing the occlusion scenario. The bottom edges of the building, L_1, L_2, L_3, and L_4, are all occluded and cannot be extracted among three aerial photos. The occlusion of bottom edges makes LSMDF difficult to determine the ground height (dZ) of model's datum point and building's height (h). However, the rooftop's height (dZ + h) can still be determined by the rooftop edges, L_5, L_6, L_7, and L_8. If either the building's height (h) or the ground height (dZ) can be obtained by other data source, a constraint can be added on the parameter so the other parameters can be solved correctly. There are two options to add constrain on parameters, (1) taking the constrained parameter's increment as a constant in the observation equations; or (2) adding a relatively large weight on the parameter's increment.

Take the regular box-like building in figure 9 as example. We set the ground height (dZ) as 40.287m and giving a large weight (9999.9) on the increment (ΔdZ), and manually fit the model approximately. The buffer size remains the same setting as the previous experiment. The model achieves optimal fitting after 26 iterations which is 4 iterations less than fitting without constraint. The optimal fitting results are shown in figure 11. The optimal model parameters are very similar to previous results, except for the building's height (h) or the ground height (dZ). There's only 0.034m difference in rooftop's height (h + dZ), which also proves the reliability of the least-squares fitting algorithm.

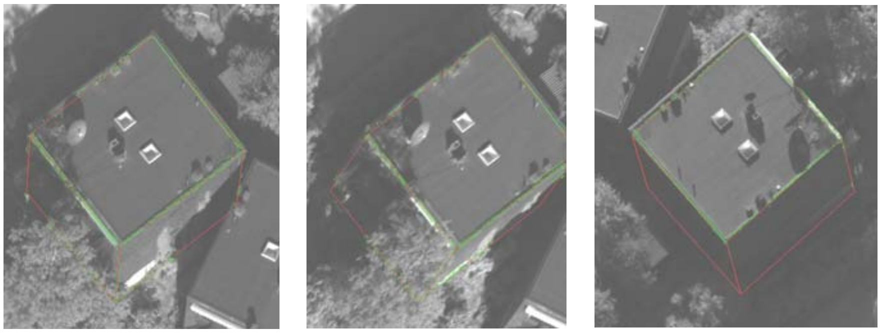


Photo 01-0013 Photo 01-0014 Photo 02-0013

$l = 11.808m$	$\sigma^2_{\Delta l} = 0.0256$
$w = 10.807m$	$\sigma^2_{\Delta w} = 0.0295$
$h = 7.622m$	$\sigma^2_{\Delta h} = 0.0001$
$\mathrm{d}X = 369353.695m$	$\sigma^2_{\Delta \mathrm{d}X} = 0.0139$
$\mathrm{d}Y=6669671.725m$	$\sigma^2_{\Delta \mathrm{d}Y} = 0.0169$
$\mathrm{d}Z = 40.320m$	$\sigma^2_{\Delta \mathrm{d}Z} = 0.0001$
$\alpha = 8.519016^\circ$	$\sigma^2_{\Delta \alpha} = 0.0003$

Edge Pixel No. within Buffer:
Photo 01-0013: 2168
Photo 01-0014: 2053
Photo 02-0013: 1387

Figure 11. The box primitive model achieves optimal fit after 26 iterations with constrained dZ.

Buffer Size

The horizontal buffer is defined along each visible edge segment projected on aerial photos. The size of the buffer will affect the pull-in range of the fitting. The wider buffer will bring more extracted edge pixels, which will not only increase the chances to include the edge pixels but also those non-edge pixels. The narrower buffer includes fewer pixels, which will decrease the chances affected by not-really-edge pixels but may fix on wrong position as well. Therefore, it would be a better strategy to start with a wider buffer and decrease the buffer size step-by-step. If the manually adjustment has approximately fit the model on all aerial photos, the initial buffer size can be smaller to improve the convergence.

Take the rectangular box-like building in figure 9 for the control group. But the initial horizontal buffer on photos is set to 0.3*mm* and still decrease 0.05mm after each iteration, until it reaches the minimum value of 0.05*mm*. The model achieves optimal fit after 5 iterations, which is 25 iterations less than starting with a 0.5*mm* buffer. The optimal fitting results are shown in figure 12. The smaller variances of parameter's increments are the proof that the narrower initial buffer size does improve the convergence.

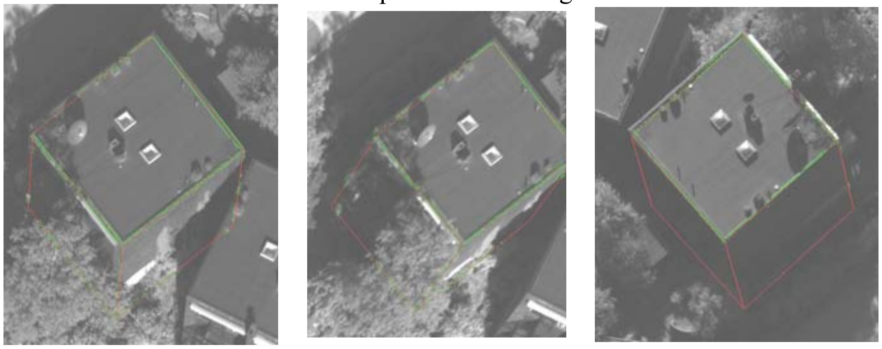


Photo 01-0013 Photo 01-0014 Photo 02-0013

$l = 11.694m$	$\sigma^2_{\Delta l} = 0.0250$
$w = 10.862m$	$\sigma^2_{\Delta w} = 0.0287$
$h = 10.007m$	$\sigma^2_{\Delta h} = 0.0001$
$\mathrm{d}X = 369353.852m$	$\sigma^2_{\Delta \mathrm{d}X} = 0.0137$
$\mathrm{d}Y = 6669671.618m$	$\sigma^2_{\Delta \mathrm{d}Y} = 0.0161$
$\mathrm{d}Z = 37.946m$	$\sigma^2_{\Delta \mathrm{d}Z} = 0.0001$
$\alpha = 8.197842°$	$\sigma^2_{\Delta \alpha} = 0.0003$

Edge Pixel No. within Buffer:
Photo 01-0013: 2256
Photo 01-0014: 2005
Photo 02-0013: 1441

Figure 12. The box primitive model achieves optimal fit after 5 iterations with constrained d*Z*.

Modeling by Using Both Aerial Photos and LiDAR Data

Since LiDAR data provides directly massive and accurate 3D points coordinates, it can not only help determining the initial value of vertical parameters but also increase the reliability of the fitting. Since the approximate horizontal range of the building has been determined by manually fitting and horizontal fitting on aerial photos, the LiDAR points belong to the building can be extracted directly. For determining the initial ground height (d*Z*), the lowest point within the 5*m* horizontal buffer zone along the building boundary is selected. The points within the building's horizontal boundary are classified by 0.5*m*-height intervals. The highest frequency class which is considered as the mode is taken as the initial value of rooftop's height (*h* + d*Z*). The initial model is then fit to aerial photos for solving the horizontal parameters and then fit to LiDAR point cloud for vertical parameters.

By setting the initial horizontal buffer as 0.5*mm* on photos and the initial vertical buffer as 1.5*m* in object space, the model achieve optimal fit after 20 iterations. It's more efficient than setting constraint on parameter d*Z*. The fitting results are shown in figure 13. Apparently from the variances of increments, benefit from the LiDAR point cloud on the rooftop, the model is fit to a better position. If the initial horizontal buffer as 0.3*mm* on photos and the initial vertical buffer as 1.5*m* in object space, the model achieve optimal fit after 5 iterations. The fitting results are shown in figure 14.

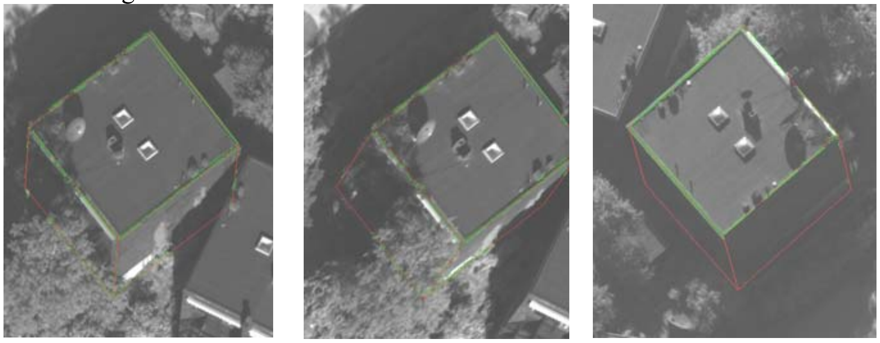
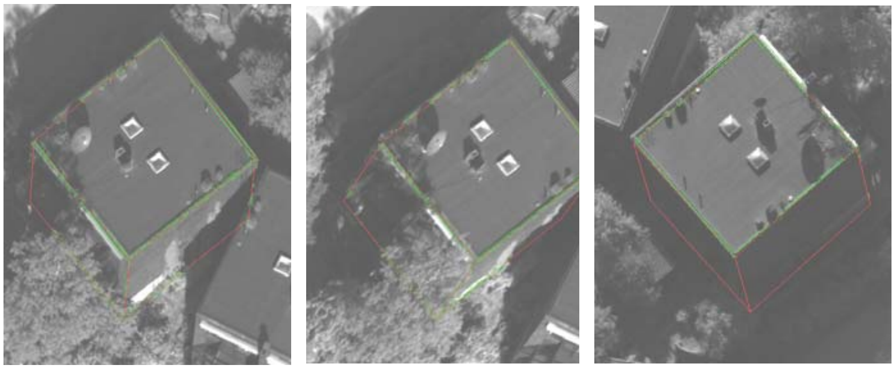


Photo 01-0013 Photo 01-0014 Photo 02-0013

$l = 11.865m$	$\sigma^2_{\Delta l} = 0.0221$
$w = 11.010m$	$\sigma^2_{\Delta w} = 0.0215$
$h = 8.175m$	$\sigma^2_{\Delta h} = 0.0001$
$\mathrm{d}X = 369353.557m$	$\sigma^2_{\Delta \mathrm{d}X} = 0.0140$
$\mathrm{d}Y = 6669671.659m$	$\sigma^2_{\Delta \mathrm{d}Y} = 0.0150$
$\mathrm{d}Z = 40.288m$	$\sigma^2_{\Delta \mathrm{d}Z} = 0.0001$
$\alpha = 8.738343°$	$\sigma^2_{\Delta \alpha} = 0.0002$

Edge Pixel No. within Buffer:
Photo 01-0013: 2499
Photo 01-0014: 2147
Photo 02-0013: 1961

Figure 13. The box primitive model achieves optimal fit after 20 iterations by using both aerial photos and LiDAR data. (Initial horizontal buffer size: 0.5*mm*)



$l = 11.696m$	$\sigma^2_{\Delta l} = 0.0251$
$w = 10.882m$	$\sigma^2_{\Delta w} = 0.0282$
$h = 7.713m$	$\sigma^2_{\Delta h} = 0.0001$
$\mathrm{d}X = 369353.828m$	$\sigma^2_{\Delta \mathrm{d}X} = 0.0142$
$\mathrm{d}Y = 6669671.625m$	$\sigma^2_{\Delta \mathrm{d}Y} = 0.0161$
$\mathrm{d}Z = 40.288m$	$\sigma^2_{\Delta \mathrm{d}Z} = 0.0001$
$\alpha = 8.3221452^\circ$	$\sigma^2_{\Delta \alpha} = 0.0003$

Edge Pixel No. within Buffer:

Photo 01-0013:	2184
Photo 01-0014:	2002
Photo 02-0013:	1498

Photo 01-0013 Photo 01-0014 Photo 02-0013

Figure 14. The box primitive model achieves optimal fit after 5 iterations by using both aerial photos and LiDAR data. (Initial horizontal buffer size: 0.3*mm*)

CONCLUSIONS & RECOMMENDATIONS FOR FUTURE WORK

A flexible 3D modeling tool called floating models is proposed as a model-based building reconstruction. Along with the tailored least-squares model-data fitting algorithm, building models can be reconstructed semi-automatically using aerial photos and LiDAR point cloud. Horizontal parameters are fit from aerial photos and vertical parameters are fit from LiDAR point cloud. According to the case study, the proposed modeling procedure goes smoother and faster with the increasing of operating experiences. Some characteristics of the proposed approach could be summarized:

1. For most of the normal buildings, floating model does increase efficiency than the conventional point-by-point stereo measurement.
2. The labor-consuming measurement is carried out by computer while the operator only has to select model type and approximately fit it.
3. The shape constraints implicitly consider the geometric nature, such as parallel edges and rectangle corners, unchanged after reconstruction.
4. It is possible to reconstruct the whole building even if a part of it is occluded or invisible.
5. Although we fit model to versatile data sources in this research, floating model is also applicable to single data source, such as only aerial photos or only LiDAR point cloud.

However, the currently proposed model types are not sufficient for dealing with every kind of buildings. To improve the completeness rate of the proposed modeling approach, we recommend designing more model types, such as buildings with cylinder and dome structure. Furthermore, if some pattern recognition techniques can be applied to detect the building model type, the automation of the proposed approach may be greatly improved.

ACKNOWLEDGEMENT

Part of this research is completed through funding from GOIDE phase IV-17, NSERC. The aerial photos and LiDAR data are provided by the EuroSDR "Registration Quality – Towards Integration of Laser Scanning and Photogrammetry" project. The first author would also like to appreciate the grant support (NSC97-2917-I-564-122) for his post-doctoral research from the National Science Council, ROC (Taiwan).

REFERENCES

Ameri, B., 2000. Feature Based Model Verification (FBMV): A New Concept for Hypothesis Validation in Building Reconstruction. *The XIXth Congress of the International Society for Photogrammetry and Remote Sensing*, Amsterdam, Netherlands, pp.24-35.

Baillard, C., A. Zisserman, and O. O. England, 1999. Automatic reconstruction of piecewise planar models from multiple views. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR '99)*, Ft. Collins, CO, pp. 2559-2565.

Berlin, 2010. 3D-Stadtmodell Berlin. Internet WebPages, http://www.3d-stadtmodell-berlin.de/3d/en/A/seite0.jsp, Last Accessed: 1st February, 2010.

Bhanu, B., D.E. Dudgeon, E.G. Zelnio, A. Rosenfeld, D. Casasent, and I.S. Reed, 1997. Guest Editorial Introduction to the Special Issue on Automatic Target Detection and Recognition. *IEEE Transactions on Image Processing*, Vol.6, No.1, pp.1-6.

Böhm, J., C. Brenner, J. Gühring, and D. Fritsch, 2000. Automated Extraction of Features from CAD Models for 3D Object Recognition. *The XIXth Congress of the International Society for Photogrammetry and Remote Sensing*, Amsterdam, Netherlands, pp.76-83.

Braun, C., T.H. Kolbe, F. Lang, W. Schickler, V. Steinhage, A.B. Cremers, W. Förstner, and L. Plümer, 1995. Models for Photogrammetric Building Reconstruction. *Computers & Graphics*, Vol.19, No.1, pp.109-118.

Brenner, C., 1999. Interactive Modelling Tools for 3D Building Reconstruction. *Photogrammetric Week '99*. Wichmann, Stuttgart, pp.23-34.

Brenner, C., 2000. Towards Fully Automatic Generation of City Models. *The XIXth Congress of the International Society for Photogrammetry and Remote Sensing*, Amsterdam, pp.85-92.

Brenner, C. 2005. Building Reconstruction from Images and Laser Scanning. *International Journal of Applied Earth Observation and Geoinformation*, Vol.6, No.3-4, March 2005, Elsevier, pp.187-198.

Das, S., B. Bhanu, and C.-C. Ho, 1996. Generic Object Recognition Using Multiple Representations. *Image and Vision Computing*, Vol.4, No.5, pp.323-338.

Englert, R. and E. Gülch, 1996. One-eye Stereo System for the Acquisition of Complex 3D Building Descriptions. *Geographic Information System*, Vol. 4, pp. 1-11.

EOS Systems, 2010. PhotoModeler - accurate and affordable 3D photogrammetry measurement and scanning. Internet WebPages, http://www.photomodeler.com/index.htm, Last Accessed: 1st February, 2010.

Ermes, P., 2000. Constraints in CAD Models for Reverse Engineering Using Photogrammetry. *The XIXth Congress of the International Society for Photogrammetry and Remote Sensing*, Amsterdam, Netherlands, pp.215-221.

Förstner, W., 1999. 3D City Models: Automatic and Semi-automatic Acquisition Methods. *Photogrammetric Week 2003*, University of Stuttgart, pp.291-303.

Grün, A., 2000. Semi-automated Approaches to Site Recording and Modeling. *The XIXth Congress of the International Society for Photogrammetry and Remote Sensing*, Amsterdam, Netherlands, pp.309-318.

Grun, A., F. Steidler, and X. Wang, 2002. Generation and Visualization of 3D-City and Facility Models Using CyberCity Modeler, *Map Asia*, 8 August 2002. CD-ROM.

Habib, A. F., E.-M. Kim, and C.-J. Kim, 2007. New Methodologies for True Orthophoto Generation. *Photogrammetric Engineering & Remote Sensing*, Vol. 73, No. 1, pp.25-36.

Habib, A. F., E. Kwak, S. Wang, and A. P. Kersting, 2009. Report of (University of Calgary): EuroSDR "Registration Quality – Towards Integration of Laser Scanning and Photogrammetry", University of Calgary, Calgary, Canada.

Henricsson, O. and E. Baltsavias, 1997. 3-D Building Reconstruction with ARUBA: A Qualitative and Quantitative Evaluation. *Automatic Extraction of Man-Made Objects from Aerial and Space Images (II)*. Monte Verita, Birkhäuser Verlag Basel, pp.65-75.

Kim, C.-J. and A. F. Habib, 2009. Object-Based Integration of Photogrammetric and LiDAR Data for Automated Generation of Complex Polyhedral Building Models. *Sensors*, Vol. 9, No. 7, pp.5679-5701.

Lang, F. and W. Förstner, 1996. 3D-City Modeling with a Digital One-eye Stereo System, *The XVIIIth Congress of the International Society for Photogrammetry and Remote Sensing*, Vienna, Austria, pp.415-420.

Lowe, D. G. 1987. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, Vol.31, No.3, pp.355-395.

Lu, Y. H., J. C. Trinder, and K. Kubik, 2006. Automatic Building Detection Using the Dempster-Shafer Algorithm. *Photogrammetric Engineering & Remote Sensing*, Vol.72, No.4, pp.395-403.

Rakkolainen, I. and T. Vainio, 2001. A 3D City Info for mobile users. *Computers & Graphics*, Vol.25, No.4, pp.619-625.

Sester, M. and W. Förstner, 1989. Object Location Based on Uncertain Models, *Mustererkennung 1989*. Informatik Fachberichte 219. Springer Verlag, pp.457-464.

Steinicke, F., K. H. Hinrichs, F., and T. Ropinski, 2006. A Hybrid Decision Support System for 3D City Planning, *International Commission II Symposium (ISPRS)*, Vienna, Austria, pp.103-108.

Suveg, I. and G. Vosselman, 2004. Reconstruction of 3D building models from aerial images and maps. *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol.58, No.3-4, pp.202-224.

Tseng, Y.-H. and S.D. Wang, 2003. Semiautomated Building Extraction Based on CSG Model-Image Fitting. *Photogrammetric Engineering & Remote Sensing*, Vol.69, No.2, pp.171-180.

van den Heuvel, F.A., 2000. Trends in CAD-based Photogrammetric Measurement. *The XIXth Congress of the International Society for Photogrammetry and Remote Sensing*, Amsterdam, Netherlands, pp.852-863.

Veldhuis, H., 1998. Performance Analysis of Two Fitting Algorithms for the Measurement of Parameterised Objects, *International Archives of Photogrammetry and Remote Sensing*, Vol.32, Part 3/1, Columbus, OH, pp.400-408.

Vosselman, G., 1999. 3D Measurements in Images Using CAD Models, *The 5th Annual Conference of ASCI*, Heijden, Netherlands, pp.449-456.

Vosselman, G. and H. Veldhuis, 1999. Mapping by Dragging and Fitting of Wire-Frame Models. *Photogrammetric Engineering & Remote Sensing*, Vol.65, No.7, pp.769-776.

Wahl, R., G. Wölfle, P. Wertz, P. Wildbolz, and F. Landstorfer, 2005. Dominant path prediction model for urban scenarios. *The German Microwave Conference 2005*, Ulm, Germany, CD-ROM.

Wang, S. and Y.-H. Tseng, 2004. Least-squares Model-image Fitting for Building Extraction from Aerial Images", *Asian Journal of Geoinformatics*, Vol.4, No.4, pp.3-12.

Wang, S. and Y.-H. Tseng, 2009. Least-Squares Model-Image Fitting of Floating Models for Building Extraction from Images, *Journal of the Chinese Institute of Engineers*, Vol.32, No.5, pp.667-677.