A Robust Approach for Automatic Registration of Aerial Images with Untextured Aerial LiDAR Data

Lu Wang and Ulrich Neumann
Computer Graphics and Immersive Technologies Laboratory
University of Southern California
{luwang, uneumann}@graphics.usc.edu

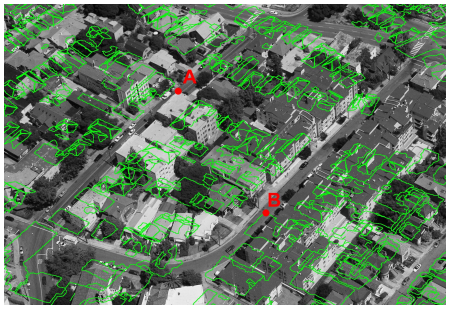
Abstract

Airborne LiDAR technology draws increasing interest in large-scale 3D urban modeling in recent years. 3D LiDAR data typically has no texture information. To generate photo-realistic 3D models, oblique aerial images are needed for texture mapping, in which the key step is to obtain accurate registration between aerial images and untextured 3D LiDAR data. We present a robust automatic registration approach. A novel feature called 3CS is proposed which is composed of connected line segments. Putative line segment correspondences are obtained by matching 3CS features detected from both aerial images and 3D LiDAR data. Outliers are removed with a two-level RANSAC algorithm that integrates local and global processing to improve robustness and efficiency. The approach has been tested on 2290 aerial images that cover a variety of urban environments in Oakland and Atlanta areas. Its correct pose recovery rate is over 98%.

1. Introduction

3D modeling of large-scale urban environments is an active research area in recent years. It has wide applications in map service, city planning, entertainment, and surveillance. The existing approaches can be divided into two kinds: imagery-based (using videos or images) [12] and LiDAR-based [7, 13]. Due to its robustness, accuracy and efficiency, airborne LiDAR technology draws increasing interest in creating large-scale 3D city models [11, 19].

Aerial LiDAR data typically does not provide texture information. To generate photo-realistic 3D models, oblique aerial images are needed for texture mapping purpose. This requires accurate registration between aerial images and 3D LiDAR data. Usually, the cameras are calibrated and an approximate camera pose for each aerial image is provided by GPS/INS systems. However, these camera parameters are often not accurate enough for precise texture mapping even with the most advanced and expensive hardware devices. In many cases, GPS/INS data is not available continuously for every frame and is distorted by significant biases and drift. Similar to [2], in this work we consider the case where there may be a large transformation between the building boundaries in aerial images and the projection of 3D building outlines according to the GPS/INS data. Figure 1 shows an example, where the green contours represent the projected 3D outlines of building rooftops detected in the LiDAR data. Point A on the 2D building boundary in the aerial image and point B on the projected 3D outline are an example of corresponding points. To refine the initial camera pose, feature correspondences between 2D aerial images and 3D LiDAR data are needed. For modeling large-scale environments, manual selection of such correspondences is expensive. Therefore, automatic aerial image to LiDAR data registration is a key component in generating photo-realistic 3D city models.



(a)

Figure 1. The green contours are the projected outlines of 3D building rooftops in the LiDAR data according to the GPS/INS data. Point B on the projected outlines and point A in the aerial image are an example of corresponding points.

There is a considerable amount of prior work on 2D image to 3D model registration. Stomas and Liu studied the registration of ground-level images with ground-level 3D LiDAR model [10, 15]. Their approach decouples camera

rotation and translation in pose estimation. Camera rotation relative to the 3D model is computed based on at least two vanishing points. In [10], camera translation is then estimated with a hypothesis-and-test scheme in matching 2D rectangles in images and parallelepipeds in 3D models. Since rectangular parallelepipeds are not always available, the authors improved their approach in [15] where the estimation of camera translation is based on line segment matching. Although this approach works well for ground-level data, it has difficulties in handling aerial images. In many cases, it is hard to detect vertical vanishing points from aerial images in which building facades are barely visible. It is also quite often that there are no dominant clusters of horizontal parallel lines in aerial images, especially for maintain areas where the distribution of buildings is irregular. These lead to inaccurate estimation of camera rotation, and further cause the failure of translation computation.

The same strategy of decoupling the estimation of camera rotation and translation is also exploited by some researchers in matching ground-level images with 3D models generated from aerial images [9, 16]. Their approaches also rely on vanishing points, and hence are not suitable for complicated aerial images. Zhao et al. worked on aligning continuous videos onto 3D point clouds [18]. Although the approach avoids the problem of plane detection from 3D point clouds, it requires structure from motion techniques for video-based 3D reconstruction which are computationally expensive and have drawbacks in accuracy and robustness. Frueh et al. proposed an approach [6] for texture mapping 3D models with oblique aerial images. Their registration method is to exhaustively search a 7-dimensional space of camera parameters so that the projected 3D model lines can be as close as possible to the 2D lines in the aerial images, which is computationally expensive.

The existing work most similar to ours is presented in [2] in which oblique aerial images are registered with untextured aerial LiDAR models. The approach utilizes the vertical vanishing point in an aerial image to estimate the pitch and roll angles of its camera rotation. The camera position and heading angle are read from GPS and compass. To refine these initial camera parameters, the features called 2D orthogonal corners (2DOCs) are extracted from both the aerial image and the digital surface model (DSM) of the LiDAR data. Each 2DOC feature corresponds to an orthogonal corner on building outlines. The 2DOCs on the DSM are projected into the aerial image according to the initial camera pose. Putative matches between the projected 2DOCs and those in the aerial image are generated by thresholding on their spatial proximity and similarity measure. The outliers are then removed with a method combining Hough transform and RANSAC. As reported in [2], the correct pose recovery rate of this approach is 91% for downtown areas but only about 50% for campus or residential areas. The main factors causing incorrect registration are: 1) failure in the extraction of 2DOC features; 2) too many outliers that cannot be handled by Hough transform or RANSAC.

We present an efficient and more robust approach for aerial image to aerial LiDAR data registration. It does not require vanishing point detection. The pitch, roll and yaw angles given by GPS/INS devices are directly used as initial camera rotation. Our main contributions are: (1) A line detection algorithm with special strategies to ensure that the line segments detected in the aerial images and those in the LiDAR data have as many matches as possible. (2) A novel feature called 3CS (3 connected segments) is introduced. Each 3CS has 3 segments connected into a chain. Compared to the 2DOC features in [2], 3CS features are more distinctive, and hence the percentage of inliers in putative feature matches is greatly increased. (3) Based on the characteristics of our problem, a two-level RANSAC algorithm is proposed in which putative feature matches are divided into multiple groups. In the first level processing, a separate RANSAC routine is applied for each group. The output is then input to a global RANSAC to remove the remaining outliers. Compared to the traditional RANSAC, the two-level scheme is more efficient, and more robust in the situations where the outliers are much more than the inliers.

The rest of the paper is organized as follows: The line segment detection algorithm is presented in Section 2. Section 3 describes the detection and description of 3CS features. In Section 4, the two-level RANSAC algorithm is introduced. Some experimental results are given in Section 5, and the paper is concluded in Section 6.

2. Line Segment Detection

Our approach requires line segment detection in aerial images and LiDAR data. In most existing 2D image to 3D model registration approaches [2, 10, 6], the following line detection algorithm is used: Edge pixels are detected with Canny detector and then linked into curves. The curves are divided into straight line segments based on thresholding on line fitting error. Although this method is efficient, it often misses the detection of many useful line segments on building outlines.

To ensure the robustness of automatic registration, the strategy in our line detector is to detect as many as possible the line segments on building outlines, even though this may increase the number of spurious segments (such as those on trees and roads). With enough useful segments, the following feature matching and RANSAC process are able to distinguish them from the spurious ones. Based on this, our line detection algorithm has the following 5 steps.

Step 1: Edge pixels are detected with the approach in [17]. Compared to Canny detector, this approach is less sensitive to the selection of thresholds. The default parameters given in [17] are used for all the aerial images in our

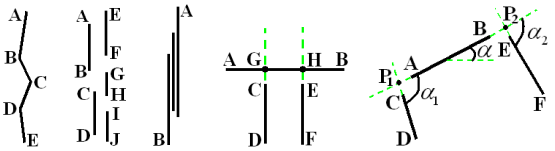


Figure 2. (a):Multi-scale polygonization. (b): Link the segments into AD and EJ. (c): Merge the parallel segments into AB. (d): Segment AB is divided into AG, GH, HB, AH and GB. (e): The description of a 3CS feature.

experiments. The edge pixels are then linked into curves based on 8-neighbor connection.

Step 2: The curves are divided into line segments with an approach similar to [1] in which a split-and-merge algorithm is used to select break points from a set of points with local extreme curvature. There are two parameters: the scale ω of the Gaussian filter in curvature computation and the threshold T on the line fitting error. Since there is no single scale or threshold suitable for all curves in all images, multiple scales and thresholds are used. The line segments obtained under different settings are all kept for the following processing. In our experiments $\omega = \{5, 11, 21, 31\}$, and the thresholds on the line fitting error are decided based on the length of the segments. Assume the length of a curve segment is L pixels, then $T = \{5, 10, ..., min(0.1L, 20)\}$. As shown in Figure 2(a), the curve can be divided into 4 line segments AB, BC, CD and DE, while the curve will be approximated with only one segment AE with a larger threshold T. The reason to keep all these segments is that there are two possible factors causing the zigzag patterns on building outlines: building structures and noise (such as the occlusion by trees). In the former case, the zigzag details are useful for the registration purpose, whereas in the latter case, only the longest segments (such as AE) can appear in both aerial images and LiDAR data.

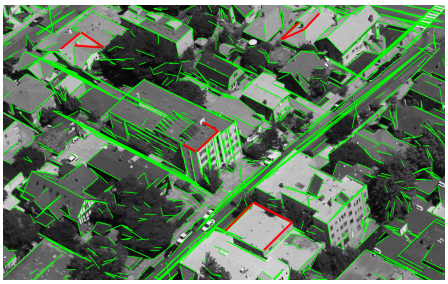
Step 3: It is often that a continuous line segment is broken into several fragments in an aerial image due to occlusion or failure in edge detection. Therefore, a linking process is needed to recover the original segment. For each segment, we search the neighborhood around each of its endpoints to find the segments that can be linked with it. As demonstrated in Figure 2(b), two segments AB and CD can be linked if the following conditions are satisfied: 1) The difference d_a of their orientation is less than a threshold (10 degrees in our experiments). 2) The distance d_h between B and C is smaller than $min(|AB|, |CD|)$. 3) The vertical distance d_v from point C to the underlying line of AB is smaller than a threshold $min(d_v, 10)$. In the case that a line segment can be linked with multiple segments, its most linkable segment is the one with the smallest value of $w_1 d_a + w_2 d_v + w_3 d_h$ ($w_1 = 1, w_2 = 0.1, w3 = 2$ in our experiments). Two segments are actually linked into a new segment only when they are the most linkable segments of each other. New segments can be further linked with other segments. As an example, for the segments in Figure 2(b), AB and CD will be linked into AD; EF, GH, and IJ will be linked into EJ. Note that the new segments and the original segments before the linking are all kept.

Step 4: As depicted in Figure 2(c), it is a common situation in urban environments that multiple parallel line segments are closely located in a narrow area in aerial images. Usually, these segments correspond to a single line segment in LiDAR data, so it is necessary to merge them. The merging process is similar to the linking process in Step 3. Two segments can be merged if they satisfy the following conditions: 1) They are almost parallel. 2) They have overlapping along their line directions. 3) Their vertical distance is smaller than a threshold. As an example, a new segment AB is generated after merging in Figure 2(c).

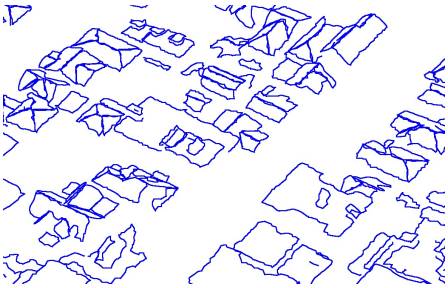
Step 5: To construct 3CS features as described in Section 3, line segments are split based on their intersection relationships. As shown in Figure 2(d), segments AB and CD intersect at point G. If the gap $|GC|$ is smaller than a threshold $T_g = 0.3 \cdot \min(|AB|, |CD|)$, AB will be split by CD into two segments AG and GB. The newly generated segments may further be split by other segments. Therefore, in Figure 2(d), AB will be divided into 5 segments: AG, GH, HB, AH and GB. The original segment AB is still kept for the following processing. In addition, the segments with their length shorter than a threshold T_L (15 pixels in our experiments) will be removed. Figure 3(a) shows an example of the line detection result, where the detected line segments are displayed in green color.

To extract line segments in 3D LiDAR data, the approach in [19] is used to detect planar facets on building rooftops. Unlike the approach based on DSM in [2] where only height differences are examined for plane detection, in [19] the normal of surfaces is also considered so the roof structures composed of slopes can be extracted. These detailed roof structures are very useful in registration, especially for residential areas where the exterior contours of rooftops are often corrupted by trees. The contours of the planar facets are then projected into aerial images according to the initial camera parameters given by GPS/INS systems (visibility will be handled with the Z-buffer technique [14]). As an example, Figure 3(b) is the projection of the rooftop outlines extracted from the LiDAR data of the area shown in Figure 3(a). These contours will then be divided into line segments with the same approach as described above in Step 2-5.

In our experiments, the resolution of the aerial images is 4992×3328 pixels. The average number of line segments detected in an aerial image is 29,000. Although many of them are spurious, most of the meaningful segments on building outlines are correctly extracted (judged visually by a human), which is important for the following registration.



(a)



(b)

Figure 3. (a): The result of line detection. Several 3CS features are shown in red color. (b): The projection of the 3D rooftop outlines extracted from the LiDAR data.

3. Detection and Description of 3CS Features

The 2DOC features proposed in [2] are not very distinctive. Each 2DOC is described by only two angles: the orientation of its two lines. This leads to a large number of outliers in the putative 2DOC matches. We introduce a more distinctive and still repeatable feature called 3CS (3 connected segments) which is similar to the kAS features applied in object detection in [4].

3.1. The Detection of 3CS Features

Each 3CS feature consists of 3 segments that are connected one after another into a chain. As an example, four 3CS features are shown in Figure 3(a) whose line segments are displayed in red color. Since the endpoints of the detected segments are often inaccurate, two segments AB and CD are regarded as connected if the following conditions are satisfied. As demonstrated in Figure 2(e), P_1 is the intersection of AB and CD. $|\overrightarrow{AP_1}|$ is the signed distance from A to P_1 in the direction of $\overrightarrow{BA}$ ($|\overrightarrow{AP_1}| < 0$ if P_1 is on the same side of B relative to A). $|\overrightarrow{CP_1}|$ is the signed distance from C to P_1 in the direction of $\overrightarrow{DC}$. Then, if $0 \leq |\overrightarrow{AP_1}| \leq 0.3|AB|$ and $0 \leq |\overrightarrow{CP_1}| \leq 0.3|CD|$, AB and CD are a pair of connected segments. In the case $|\overrightarrow{AP_1}| < 0$, AB will be divided into two sub-segments in Step 5 discussed in Section 2, and these sub-segments will be checked if they are connected with CD.

To detect 3CS features from an aerial image, for each line segment denoted as AB, we search the neighborhood of its two endpoints to find the line segments that are connected with it. Assume EF is one of the segments that is connected with AB at point A, and GH is connected with AB at point B. The three segments AB, EF and GH form a 3CS feature. AB is called the central segment and the middle point of AB is called the center of the 3CS feature. With the same approach, 3CS features are also extracted from 3D LiDAR data based on the line segments detected on the projected outlines of building rooftops.

In practice, to reduce the number of 3CS features detected in an aerial image, a 3CS will be removed if: 1) The central segment is almost parallel to the other two segments; or 2) The length ratio between any two segments is larger than a threshold (7 in our experiments). Such a 3CS feature is unlikely to be composed of segments on building outlines. In our experiments, the average number of 3CS features detected in an aerial image (4992×3328 pixels) is about 150,000.

Note that unlike some object recognition systems such as [3] in which the number of line segments in a segment group can be arbitrarily large, the number of segments in a 3CS feature is limited to be 3. Increasing this number will decrease the repeatability of the features, and greatly increase their number and the computation in the following registration process.

3.2. The Description of 3CS Features

For a 3CS feature composed of segments AB, CD and EF, with AB as the central segment, it can be described with 6 attributes $(l, l_1, l_2, \alpha, \alpha_1, \alpha_2)$ as demonstrated in Figure 2(e). P_1 and P_2 are the two intersection points of AB with CD, and AB with EF respectively. Then, $l = |P_1P_2|$, $l_1 = |DP_1|/|P_1P_2|$, and $l_2 = |FP_2|/|P_1P_2|$. α is the angle from the vector $\overrightarrow{AB}$ to the X axis. α_1 is the angle from $\overrightarrow{CD}$ to $\overrightarrow{AB}$, and α_2 is the angle from $\overrightarrow{EF}$ to $\overrightarrow{AB}$.

To measure the dissimilarity of two 3CS features with the description of $(l, l_1, l_2, \alpha, \alpha_1, \alpha_2)$ and $(l', l_1', l_2', \alpha', \alpha_1', \alpha_2')$ respectively, the following equation is applied:

$$D = \begin{cases} \sum_{k=1}^{6} d_k, & \text{if } d_{1\sim 6} < 1; \\ \infty, & \text{else,} \end{cases} \tag{1}$$

where,

$$\begin{cases} d_1 = \frac{\max(l,l')/\min(l,l')-1}{T_1}; \\ d_2 = \frac{\max(l_1,l_1')/\min(l_1,l_1')-1}{T_2}; \\ d_3 = \frac{\max(l_2,l_2')/\min(l_2,l_2')-1}{T_2}; \\ d_4 = \frac{|\alpha-\alpha'|}{T_3}; \\ d_5 = \frac{|\alpha_1-\alpha_1'|}{T_4}; \\ d_6 = \frac{|\alpha_2-\alpha_2'|}{T_4}. \end{cases} \tag{2}$$

D is the dissimilarity value of the two 3CS features. A smaller D means the two 3CSs are more similar. In Eq.2, $T_{1\sim4}$ are thresholds. $d_{1\sim6}$ are the normalized dissimilarities of the two 3CS features with respect to each of the 6 attributes. If the difference of one of the attributes is larger than the corresponding threshold (represented by $d_{1\sim6} < 1$ in Eq.1), the two 3CS features are not likely to be matched, and so their dissimilarity D is set ∞.

The thresholds $T_{1\sim4}$ decide the size of the searching space in looking for the putative 3CS matches. In our experiments, $T_{1\sim4}$ are 1, 0.5, 45° and 30° respectively. Note that T_2 and T_4 are smaller than T_1 and T_3 respectively. This is because the variations of the relative length ratio and angle between neighboring line segments are usually smaller than those of the absolute length and orientation.

For each projected 3CS feature f of the LiDAR data, its putative corresponding 3CS features in an aerial image are found in a circular neighborhood of its center, whose dissimilarity values with f are smaller than ∞. If multiple such 3CS features are found, only the best two with the smallest dissimilarity values are kept.

Since the number of 3CS features in an aerial image is usually very large, to improve the speed in searching for putative 3CS matches, an index structure is created by dividing the aerial image into a grid and splitting the range of each attribute of the 3CS descriptor into bins. For each projected 3CS feature of the LiDAR data, the buckets that possibly contain its putative corresponding 3CSs in the aerial image can be computed.

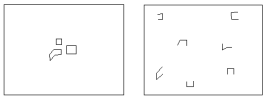
4. Two-level RANSAC Algorithm

In most cases, we can make the same assumption as in [2] that the error in the camera location given by a GPS device is very small relative to the distance from the camera to the buildings. Therefore the transformation between the projected 3CSs of the LiDAR data and the 3CSs in the aerial image can be considered to be purely caused by camera rotation, and hence is a homography.

RANSAC can be used to remove outliers from putative feature matches [2, 5]. However, due to the large size of the aerial images and the huge number of putative matches, direct applying the traditional RANSAC approach has the following problems. **Problem (a)**: When the outliers are much more than the inliers, some outliers can be fit with a homography accidentally. If the number of these outliers is larger than that of the real inliers, it will lead to completely wrong registration. **Problem (b)**: Due to image radial distortion or because the error of the initial camera location cannot be completely ignored, sometimes the underlying transformation is not a strict homography. This makes it hard to choose a good threshold on the homography fitting error for selecting inliers. If the threshold is small, many of real inliers will be removed. If it is large, many of the outliers will be included as inliers and harm the accuracy of the camera pose estimation. **Problem (c)**: Although two 3CS matches provide enough constraints to decide a homography (as discussed later), the homography estimated with two 3CS matches is usually not accurate enough to describe the transformation of the whole image since the constraints are limited in a small part of the image. Therefore, more than two 3CS matches should be sampled at each RANSAC iteration, which will greatly increase the required number of iterations and make the computation expensive.

We present a two-level RANSAC algorithm. Instead of processing all the putative 3CS matches as a whole, they are divided into groups so that the 3CS features in each group are contained in a local area. During the first-level processing, a separate RANSAC routine is run for each group, and the selected inliers are called qualified 3CS matches which are input to the second level processing where a global RANSAC is applied. Rather than selecting individual 3CS matches, the global RANSAC operates based on groups and decides which groups are inliers. The qualified 3CS matches of these groups are output as the final matches. The reasons that this two-level scheme can handle the aforesaid problems of the traditional RANSAC are analyzed below.

For Problem (a), our approach not only considers their number but also their spatial distribution in evaluating if a group of feature matches are inliers. This is illustrated in Figure 4 where the feature matches depicted in Figure 4(a) (only the projected 3CS features of LiDAR data are represented) are more likely to be inliers than those in Figure 4(b) since they are close to each other, even though the numbers of matches are the same in the two cases. This is because the features in a small area are less than those in a large area so the possibility that the same number of features satisfying a homography are caused by random noise is smaller if they are clustered in a small area than if they are distributed sparsely over the image. Another reason is that buildings usually form blocks or clusters, which is important knowledge for distinguishing inliers and outliers for areas with heavy vegetations.



(a) (b)

Figure 4. (a):Closely located feature matches. (b): Sparsely distributed feature matches.

In our approach, the above strategy is implemented in the second level RANSAC by giving different groups different weights. At each of its iterations, assume there are p groups selected as inliers, and the number of qualified 3CS matches in the k-th group is n_k. Then, the score of each iteration is

calculated with:

$$G = \sum_{k=1}^{p} (n_k \cdot \sqrt{n_k}), \tag{3}$$

where $\sqrt{n_k}$ serves as the weight for group k. Because of this weight, the situation where the qualified 3CS matches concentrate in a few groups will get higher score than the situation where they are uniformly spread among all of the groups, even if the total numbers of these 3CS matches in both situations are the same.

The two-level scheme also has advantage with regard to Problem (b). Although the transformation in the whole image may not be a strict homography so a larger threshold T_2 on the homography fitting error should be used in the global RANSAC, the transformation in a local area can be more accurately described with a homography. Thus, the threshold T_1 used in the first level RANSAC for each group can be set smaller than T_2. This double-threshold strategy can weaken the dependence on the assumption of strict homography in the whole image while remove the 3CS matches that are not compatible with their neighbors.

Finally, the two-level RANSAC algorithm is computationally efficient. Only two 3CS matches are sampled at each iteration of the first level RANSAC. This is because the area of each 3CS group is much smaller than the whole image, and the homography computed from two 3CS matches is usually accurate enough to describe the transformation inside the local area. The second level RANSAC operates based on groups. At each iteration, three groups are sampled and all of their qualified 3CS matches are used to estimate a homography. Since the number of groups is much smaller than the total number of putative 3CS matches, the required number of interactions in the second level RANSAC is not large.

The details of the two-level RANSAC algorithm are described as follows:

Step 1: The putative 3CS matches are divided into groups according to spatial proximity. In doing this, an aerial image is divided into windows. The window size is set $s \times s$ pixels ($s = 1000$ in our experiments). Start from the left top corner of the image, a window is shifted from left to right and top to bottom. The step size is $s/4$ so that neighboring windows have overlapping. For each window, if the number of projected line segments of the LiDAR data inside it is larger than m, it will be split into four sub-windows. A sub-window will be further split until the number of projected segments inside it is smaller than m. The projected 3CS features inside each window or sub-window form a group. Each projected 3CS of the LiDAR data may have one or two putative corresponding 3CSs in the aerial image (see Section 3). Therefore, each window or sub-window defines a group of putative 3CS matches. The number m controls the size of a group. In our experiments, the average number of projected line segments of the LiDAR data in an aerial image is 3500. From experiments, we found $m = 50$ is a balanced choice, so that each image has about 300 groups. Note that different groups may have overlapping.

Step 2: Assume that a 3CS feature has segments AB, CD and EF, with AB its central segment, and the segments in its corresponding 3CS are $A'B'$, $C'D'$ and $E'F'$. Since the endpoints of the detected line segments are often inaccurate, they cannot be used to estimate the homography. However, the two intersection points P_1 and P_2 as shown in Figure 2(e) are more precise, and each correspondence of such points provides two linear constraints on the homography. In addition, the orientation of the line segments is also reliable. The fact that point D (or F) after the homography transformation is on line $C'D'$ (or $E'F'$) gives a linear constraint. Therefore, each 3CS feature match provides 6 linear constraints, and two matches are enough to completely decide the homography [8].

For each group of putative 3CS matches, a separate RANSAC routine is applied to remove its outliers. At each iteration, two 3CS matches are uniformly sampled, from which a homography is then estimated. The other putative matches are regarded as inliers if their homography fitting errors are smaller than a threshold T_1 (The homography fitting error of the line constraint is the vertical distance from the transformed point to the line). In our experiments, $T_1 = 10$ pixels. The inliers selected from each group are called qualified 3CS matches.

Step 3: A global RANSAC is applied based on groups. At each of its iterations, three groups of qualified 3CS matches are uniformly sampled, from which a homography is estimated. If any of the homography fitting errors of these 3CS matches is larger than a threshold T_2, a new sample of three groups will be generated. Otherwise, each of the other groups will be examined to see if the homography fitting errors of its qualified 3CS matches are all smaller than T_2. If so, it will be selected as an inlier. The score at each iteration is computed with Eq.3. When the RANSAC terminates, the groups selected as inliers at the iteration with the highest score are returned together with the homography computed based on the qualified 3CS matches in these groups. In our experiments, the threshold $T_2 = 25$ pixels, which is larger than $T_1 = 10$ used in Step 2. The reason has been given previously in this section.

Step 4: For the groups identified as inliers in Step 3, their qualified 3CS matches are kept in the final list of the correct 3CS matches. For the other groups, all of their putative 3CS matches will be checked again against the homography returned at Step 3, and those with the homography fitting error smaller than T_1 will also be kept. The reason that the lower threshold T_1 is used here instead of T_2 is that these

matches are isolated and have less confidence to be correct unless they can satisfy the homography more strictly.

Based on the corresponding intersection points in the obtained 3CS matches, camera parameters can be estimated and refined with the approach in [8]. The approach in [6] is then used for texture mapping with multiple aerial images.

5. Experimental Results

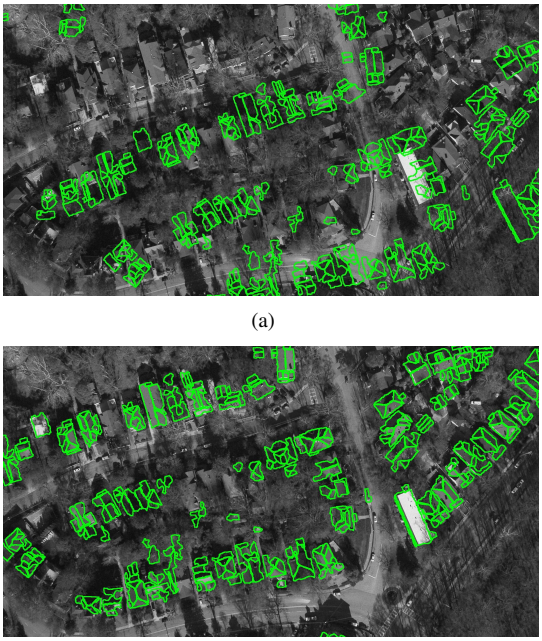
We did extensive experiments to examine the proposed system. Two datasets are tested. The first one consists of 306 oblique aerial images covering a 1.5×1.4 km^2 area in the city of Oakland. The second one has 1984 oblique aerial images covering a 2.7×2.6 km^2 area in the city of Atlanta. As in [2], the urban environments in these two datasets can be classified into three types: downtown, campus and residential areas. In downtown areas, buildings are tall and dense while trees are sparse. In regions such as campus, large buildings are sparsely distributed in dense trees. In residential areas, houses are usually short and small, and located among dense vegetation.

The correctness of the automatic registration is evaluated by checking the average distance between the corresponding points manually labeled on the aerial images and the building outlines in the LiDAR data projected according to the refined camera parameters. It is also validated by visually examining the quality of the texture mapping.

For all of the 306 aerial images in the first dataset, our system accurately recovered their camera parameters. For the second dataset, 1951 aerial images are correctly registered with the 3D LiDAR data while the registration for the other 33 images is wrong. Therefore, the overall correct pose recovery rate of our approach is 98.5%. The images that cannot be correctly registered are mainly from residential areas where most of the buildings are seriously occluded by trees. In these situations, the detection of planar facets in the 3D LiDAR data and the detection of line segments on building outlines in aerial images become very difficult.

Figure 5 shows an example of the alignment between the aerial image and the projection of 3D building outlines (the green contours) with the initial camera pose (Figure 5(a)) and with the refined camera pose after the automatic registration (Figure 5(b)). Figure 6(a)-(d) are several screen shots of the textured 3D models. To make the 3D models look cleaner, most of the trees are removed in the LiDAR data with the approach in [19].

To prove that 3CS features are more distinctive than the 2DOC features in [2], we computed the average percentage of inliers in the putative feature matches. It is 19% for 3CS features, higher than that of 2DOC features which is 4% according to [2]. In average, the approach takes about 1 minute on a PC with a 3 GHz CPU to register a 4992×3328 aerial image. Most computation is spent on the line detection. The RANSAC process takes only several seconds.



(a)

(b)

Figure 5. (a):The alignment between the aerial image and the 3D outlines of building rooftops projected with the initial camera pose. (b): The alignment with the refined camera pose.

6. Conclusion

We have presented an approach for automatic registration of aerial images with untextured aerial LiDAR data. Several strategies are taken to improve the robustness of line segment detection. A novel feature called 3CS that is more distinctive than a single line segment or the 2DOC feature in [2] is used, which greatly increases the percentage of inliers in the putative feature matches. Finally, a two-level RANSAC algorithm is proposed that is more robust and efficient than the traditional RANSAC approach in our situations where the number of putative feature matches is very large while the percentage of inliers is low, and the underlying transformation is not a strict homography.

Compared to existing approaches, our system is more robust. Its overall correct pose recovery rate is above 98%. To further improve the approach, more efficient and robust algorithms of line detection, and planar facet detection in the LiDAR data should be developed. Building detection and segmentation with high-level knowledge are also helpful.

7. Acknowledgments

This research was supported in part by CiSoft project sponsored by Chevron. We thank Airborne 1 for providing us LiDAR data and aerial images. We thank Qianyi Zhou for sharing his code of building outline extraction in LiDAR data. We appreciate useful discussions with Suya You.

(a)

(b)

(c)

(d)

Figure 6. Screen shots of textured 3D models.

References

[1] N. Ansari and E. J. Delp. On detecting dominant points. *Pattern Recognition*, 24(5):441–451, 1991.

[2] M. Ding, K. Lyngbaek, and A. Zakhor. Automatic registration of aerial imagery with untextured 3d lidar models. *CVPR*, pages 1–8, 2008.

[3] D.Jacobs. Groper: A grouping based object recognition system for two-dimensional objects. *IEEE Workshop on Computer Vision*, pages 164–169, 1987.

[4] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *PAMI*, 30(1):36–51, 2008.

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing*, 24(6):381–395, 1981.

[6] C. Frueh, R. Sammon, and A. Zakhor. Automated texture mapping of 3d city models with oblique aerial imagery. *3DPVT*, pages 396–403, 2004.

[7] C. Fruh and Z. Zakhor. An automated method for large-scale, ground-based city model acquisition. *IJCV*, 60(1):5–24, 2004.

[8] R. I. Hartley and A. Zisserman. Multiple view geometry in computer vision, 2004.

[9] S. C. Lee and R. Nevatia. Automatic integration of facade textures into 3d building models with a projective geometry based line clustering. *Computer Graphics Forum(Euro Graphics)*, 21(3):511–519, 2002.

[10] L. Liu and I. Stamos. Automatic 3d to 2d registration for the photorealistic rendering of urban scenes. *CVPR*, pages 137–143, 2005.

[11] B. C. Matei, H. S. Sawhney, S. Samarasekera, J. Kim, and R. Kumar. Building segmentation for densely built urban regions using aerial lidar data. *CVPR*, pages 1–8, 2008.

[12] M. Pollefeys, D. Nister, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3d reconstruction from video. *IJCV*, 78(2-3):143–167, 2008.

[13] C. Poullis, S. You, and U. Neumann. Rapid creation of large-scale photorealistic virtual environments. *IEEE Virtual Reality*, pages 153–160, 2008.

[14] P. Shirley, M. Ashikhmin, M. Gleicher, S. Marschner, E. Reinhard, K. Sung, W. Thompson, and P. Willemsen. Fundamentals of computer graphics. *Wellesley, MA: A K Peters, Ltd.*, 2002.

[15] I. Stamos, L. Liu, C. Chen, G. Wolberg, G. Yu, and S. Zokai. Integrating automated range registration with multiview geometry for the photorealistic modeling of large-scale scenes. *IJCV*, 78(2-3):237–260, 2008.

[16] L. Wang, S. You, and U. Neumann. Semiautomatic registration between ground-level panoramas and an orthorectified aerial image for building modeling. *ICCV Workshop on Virtual Representation and Modeling of Large-scale Environments*, pages 8–15, 2007.

[17] L. Wang, S. You, and U. Neumann. Supporting range and segment-based hysteresis thresholding in edge detection. *ICIP*, 2008.

[18] W. Zhao, D. Nister, and S. Hsu. Alignment of continuous video onto 3d point clouds. *PAMI*, 27(8):1305–1318, 2005.

[19] Q. Y. Zhou and U. Neumann. Fast and extensible building modeling from airborne lidar data. *ACM GIS*, pages 1–8, 2008.