Automatic registration of aerial imagery with untextured 3D LiDAR models

Min Ding, Kristian Lyngbaek and Avideh Zakhor
University of California, Berkeley
Electrical Engineering and Computer Science Department
{dingm, kristian, avz}@eecs.berkeley.edu

Abstract

A fast 3D model reconstruction methodology is desirable in many applications such as urban planning, training, and simulations. In this paper, we develop an automated algorithm for texture mapping oblique aerial images onto a 3D model generated from airborne Light Detection and Ranging (LiDAR) data. Our proposed system consists of two steps. In the first step, we combine vanishing points and global positioning system aided inertial system readings to roughly estimate the extrinsic parameters of a calibrated camera. In the second step, we refine the coarse estimate of the first step by applying a series of processing steps. Specifically, We extract 2D corners corresponding to orthogonal 3D structural corners as features from both images and the untextured 3D LiDAR model. The correspondence between an image and the 3D model is then performed using Hough transform and generalized M-estimator sample consensus. The resulting 2D corner matches are used in Lowe's algorithm to refine camera parameters obtained earlier. Our system achieves 91% *correct pose recovery rate for 90 images over the downtown Berkeley area, and overall* 61% *accuracy rate for 358 images over the residential, downtown and campus portions of the city of Berkeley.*

1. Introduction

3D models are needed in many applications such as urban planning, architecture design, telecommunication network design, cartography and virtual fly/drive-through. Due to their significance and vast potential, fast and automated model reconstruction has drawn a great deal of attention and effort from many researchers in the last three decades. However, most existing approaches lack either accuracy or the scalability needed for creating textured models of large urban areas. Even though model geometry can be quickly and automatically generated from aerial images or Light Detection and Ranging (LiDAR) data [15], registering imagery with a 3D model for texture mapping purposes is considerably less automated and more time-consuming; this is due to lengthy manual correspondence between a 3D model and images, or computationally intensive automated pose recovery algorithms. For instance, the pose recovery algorithm proposed by Frueh et al. [6] for texture mapping oblique aerial imagery onto untextured 3D models takes approximately 20 hours per image. In this paper, we develop a fast, automated camera pose recovery algorithm for texture mapping oblique aerial imagery onto pre-existing untextured 3D models obtained via various sensing modalities such as LiDAR. We believe that such algorithms are the key to fast, automated 3D airborne modeling of large scale environments.

There have been a number of approaches to automated texture mapping of 3D models. Stomas and Liu have developed texture mapping for ground based multiview images [11, 17]. They use vanishing points and rectangular parallelepipeds on building facades for matching features between LiDAR data and images in order to identify camera parameters. They further refine the parameters by 3D point cloud correspondence between the LiDAR data and the sparse point cloud generated from multiview geometry. Their algorithm requires multiview imagery, and runs into difficulties if there are no two pairs of correctly matched parallelepipeds. They also take advantage of the ground based image acquisition where clear parallelism and orthogonality of building contours are visible with little occlusion. By contrast in our application, it is desirable to develop an approach which uses single view, and can handle complex urban scenes with significant occlusions.

In video based texture mapping, Hsu et al. first use the tracked features for inter-frame pose prediction, and refine the pose by aligning the projected 3D model lines to those in images [8]. Neumann et al. follow a similar idea by implementing an extended Kalman filter to perform inter-frame camera parameter tracking using point and line features [14]. Both methods can lose track in situations with large pose prediction error due to occlusions. Zhao et al. instead use iterative closest point algorithm to align the point cloud from video to that from a range sensor [20]. However,

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

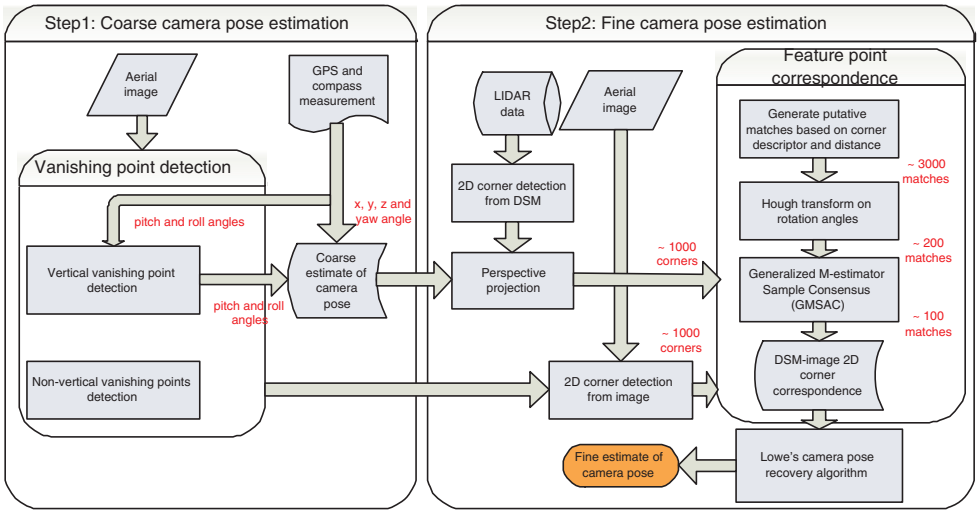


Figure 1. Camera registration system overview.

it is computationally expensive to generate 3D point clouds from a video.

Lee and Nevatia et al. use vanishing points and random sample consensus (RANSAC) based 3D-2D line pair match to find single view camera pose [10]. However, they only deal with ground based images for a single building where clear parallelism and orthogonality of building contours are visible with little occlusion. Recently, Hu et al. have created a system capable of aerial and ground based image mapping [9]. However it requires human interactions in many places such as building contour extraction from aerial images and manual point correspondence to align aerial images to LiDAR data.

In this paper, we describe a fast, automated, camera pose recovery algorithm for texture mapping oblique aerial imagery onto 3D geometry models obtained via LiDAR. In doing so, we take advantage of vanishing points and develop a feature matching technique based on 2D corners associated with orthogonal 3D structural corners. Our approach is two order of magnitude more efficient than [4] in that it can recover a camera pose in approximately 3 minutes on today's personal computers.

We assume the intrinsic camera parameters such as focal length to be fixed during the entire data acquisition process. Our approach is to tackle the camera registration problem in two steps as depicted on Fig.1. In the first step, we obtain coarse camera parameters which are further refined in the second step. We use the global positioning system aided inertial system, NAV420CA from Crossbow, to obtain coarse estimate of camera position, and its heading angle. The pitch and roll angles of the camera's rotation are then coarsely estimated from the position of the vanishing point of vertical lines in the 3D space. In the first step, vanishing points corresponding to non-vertical lines are also detected to be used for 2D corner extraction from images in the second step.

The second step of our proposed approach uses 2D corners corresponding to orthogonal structural corners in the 3D space as features. For brevity, we refer to these as 2D orthogonal corners or 2DOCs. 2DOCs are extracted from a digital surface model (DSM) obtained via LiDAR data processing [5], as well as from aerial images based on the orthogonality information implied by the vanishing points detected earlier. After projecting 2DOCs from the DSM using the coarse camera parameters obtained in the first step, putative matches between DSM 2DOCs and image 2DOCs are generated based on distance and corner descriptors' similarity. We apply Hough transform to screen out majority of the spurious matches, followed by generalized M-estimator sample consensus (GMSAC), to identify the correct DSM-image 2DOC matches. Finally, we apply Lowe's camera pose recovery algorithm [12] to the remaining 2DOC pairs in order to obtain the refined camera parameters for texture mapping.

The outline of this paper is as follows. Section 2 describes use of vanishing point and appropriate hardware readings to arrive at a coarse estimate of extrinsic camera parameters. Section 3 describes the processing chain for extracting, matching, and pruning features to refine the coarse camera pose obtained in Section 2. Section 4 examines the performance of the proposed system on 358 aerial images taken over $3.4km^2$ area in Berkeley. We conclude with future directions in Section 5.

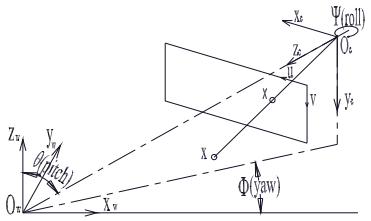


Figure 2. Definition of the extrinsic camera parameters.

2. Coarse Camera Parameter Acquisition

A calibrated camera model shown in Fig.2 is first assumed:

$$\lambda x = [\mathbf{R} \quad \mathbf{T}]X \tag{1}$$

where $x = [u, v, 1]^T$ is the coordinate on the image plane of $X = [x_w, y_w, z_w, 1]^T$ in the 3D space after perspective projection. $\mathbf{R}$ is the relative rotation matrix, $\mathbf{T}$ is the relative position matrix from the world coordinate, O_W, to the camera coordinate, O_C as shown in Fig.2, and λ is a scale factor. With the yaw (ϕ), pitch (θ) and roll (ψ) defined in Fig.2, $\mathbf{R}$ is given by:

$$\mathbf{R} = \begin{bmatrix} -c\psi s\phi + s\psi c\phi c\theta & c\psi c\phi + s\psi s\phi c\theta & -s\psi s\theta \\ s\psi s\phi + c\psi c\phi c\theta & -s\psi c\phi + c\psi s\phi c\theta & -c\psi s\theta \\ -s\psi c\phi & -s\theta s\phi & -c\theta \end{bmatrix} \tag{2}$$

where c stands for cosine and s stands for sine.

2.1. Vertical vanishing point detection for pitch and roll estimation

Under the assumption of a pin hole camera projection shown in Eqn. (1), it can be shown that a set of parallel lines in a 3D space is projected onto a set of lines intersecting at a common point on the image plane. This point is referred as a vanishing point. To obtain a coarse estimate of the pitch and roll angles of a camera, we use the vertical vanishing point corresponding to vertical lines in the 3D models. The vertical vanishing point is a consistent measurement since no matter how buildings are aligned, their vertical contour lines are almost always parallel to each other. We find the vertical vanishing point with a method similar to Gaussian sphere approach [1]. A Gaussian sphere acting as a memory, is a unit sphere with its origin at O_c, the origin of the camera coordinate. Each line segment with O_c forms a plane intersecting the sphere to create a great circle. These great circles are stored on the Gaussian sphere. It is assumed that the maximum count on the sphere represents the direction shared by multiple line segments, and is a vanishing point. The identified vertical lines on an aerial image from downtown Berkeley are highlighted in blue in Fig.3. Given that the vertical lines in the world reference

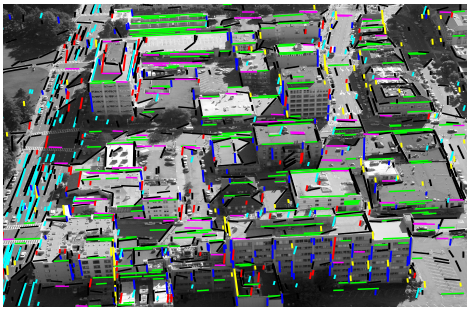


Figure 3. Extracted line segments are colored according to their perspective vanishing points. The dark blue lines are the vertical lines corresponding to the vertical vanishing point and the remaining lines correspond to non-vertical vanishing points.

space are represented as $e_z = [0, 0, 1, 0]^T$ in homogeneous coordinates, the vertical vanishing point, v_z can be shown to assume the last column of $\mathbf{R}$ based on Eqn. (1); specifically:

$$\lambda v_z = [-sin\psi sin\theta, -cos\psi sin\theta, -cos\theta]^T \tag{3}$$

From Eqn. (3), the pitch and roll angles and the scale factor can be easily computed. Together with readings of the position and heading angle from NAV420CA, we now have a set of coarse estimates of all the camera parameters to be used in the second step.

2.2. Non-vertical vanishing point detection

Even though non-vertical vanishing points are not used for the coarse camera pose estimation, they are useful for detecting 2DOCs in images as described in Section 3.2. Our proposed algorithm for extracting non-vertical vanishing points is summarized in Fig.4 and can be described as follows. 1) Bin all the line segments according to their angles, except the ones corresponding to the vertical vanishing point. 2) Examine all the line segments in a bin with the highest frequency. Identify a seed vanishing point where most of the segments in that bin intersect. The location of the vanishing point is refined by choosing the right singular vector with the least significant singular value of $\mathbf{WL}$ where $\mathbf{W}$ is a weighting square diagonal matrix with its diagonals as the lengths of the segments, and $\mathbf{L}$ stores the co-images [1]. 3) Identify more segments passing near the seed vanishing point in the two adjacent bins. The location of the vanishing point is refined again using segment length weighted singular value decomposition. 4) Repeat Step 3 by examining the segments in the next two adjacent bins until the angle spread across the two farthest bins is above a certain threshold. 5) Refine the location of the vanishing point by Levenberg-Marquardt minimization on the sum of the distances between the selected segments and the closet lines

[1] Co-image is the cross product of two vectors from the camera origin to the two endpoints of a segment.

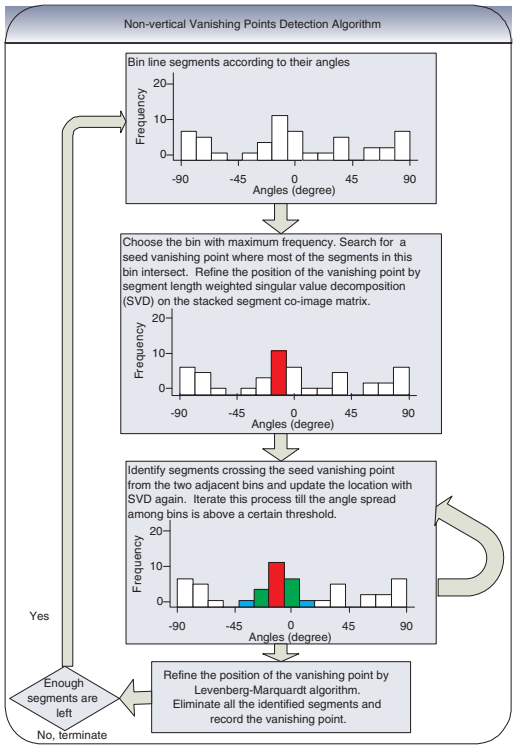


Figure 4. Non-vertical vanishing point detection algorithm flow chart.

passing through the vanishing point [7]. Remove all the selected segments and go back to Step 1 if there are enough number of segments left. The non-vertical vanishing lines obtained using this algorithm for the same sample image are shown in Fig. 3.

Since most existing techniques aim to find intersections among as many line segments as possible in an image, they fail in complex urban settings with many buildings whose alignments are not necessarily parallel [16]. This results in the intersections in the 3D space to be falsely classified as vanishing points. Our proposed algorithm overcomes this problem in several ways. First, we initialize the seed vanishing point in a bin where all the segments share similar angles. This avoids choosing a seed vanishing point which is an actual intersection in the 3D space. This preference is also reinforced by only considering segments whose slope angle difference is less than the angle spread threshold. We also use a segment length weighted singular value decomposition to favor longer segments since they bear less uncertainty in their orientation. Finally, since the identified segments are eliminated after each iteration, the convergence of the algorithm is guaranteed without a priori knowledge on the number of the vanishing points.

3. Camera parameter refinement

In this section, the set of coarse camera parameters obtained in ths first step is refined to achieve sufficient accuracy for texture mapping purposes. We employ correspondence based on 2DOC features during this process. In our application, 2DOCs correspond to orthogonal structural corners where two orthogonal building contour lines intersect. These corners are unique to urban environments, and are limited in number. It might be intuitively appealing to use true 3D corners where three orthogonal lines intersect. However we have empirically found that it is difficult to identify sufficient number of 3D corners from images given the non-ideal line segment extraction. Therefore, we have opted to relax our constraint of three mutually orthogonal line segments to two. Naturally this leads to many more false structural corners extracted from images. In the remainder of the paper, we show that these false corners can be eliminated by feature descriptors, Hough transform and GMSAC. Specifically, we show that the price paid by using 2DOC is well compensated by greater number of correct structural corners from images.

3.1. 2DOC extraction from DSM

DSM is a depth map representation of a 3D model which we assume to have been obtained from LiDAR data. To extract 2DOCs from DSM, building contours are extracted from a DSM with a region growing approach based on thresholding on height differences [5]. Due to the limited resolution of LiDAR data and inevitable noise, the contours tend to be jittery. To straighten jittery edges, we use Douglas-Peucker (DP) line simplification algorithm [3] for its intrinsic ability to preserve the position of 2DOCs since structural corners tend to correspond to the extreme vertices in a contour. Once the outer contour of each region is simplified, a simple thresholding is performed on the lengths of the two intersecting line segments and the intersection angle to identify 2DOCs. The 2DOCs are then projected back onto the image plane with the coarse camera parameters in the first step. The identified 2DOCs from the DSM corresponding to the previous sample image are shown on Fig.5 with 1548 corners in total.

3.2. 2DOC extraction from aerial images

Since vanishing points represent directions of the corresponding groups of line segments in a 3D space, the orthogonality between each pair of vanishing points also implies the orthogonality between the two groups of line segments even though they might not appear to be orthogonal in images. The endpoints of line segments belonging to orthogonal vanishing point pairs are examined to identify potential 2DOC candidates based on proximity. The resulting corners extracted from the sample image using this procedure

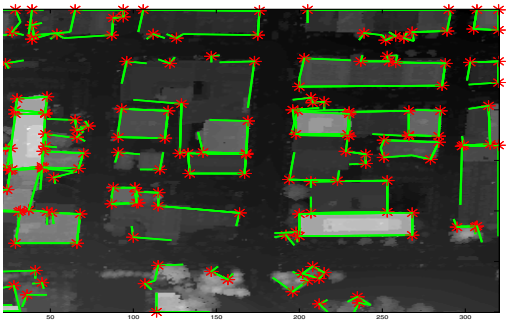


Figure 5. 2DOCs extracted from a DSM: red * denote the 2DOCs and green lines denote the two corresponding orthogonal lines.

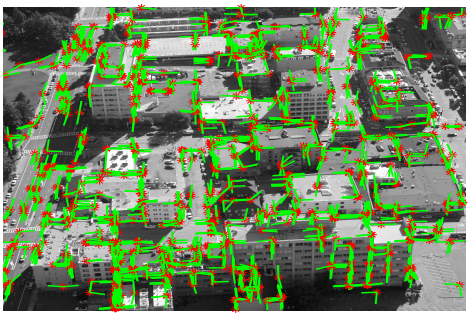


Figure 6. 2DOCs extracted from an aerial image: red * denote the 2DOCs and green lines denote the two corresponding orthogonal lines.

are shown in Fig.6 with 1099 corners in total.

3.3. 2DOC putative match generation

For every 2DOC, we define a feature descriptor consisting of the two intersecting lines' angles ($[\theta_1, \theta_2]^T$) with respect to u axis in the image plane. This descriptor is used to generate 2DOC putative matches based on thresholding on proximity and descriptors' similarity. For descriptors' similarity, we opt to use Mahalanobis distance:

$$d(x^d, x^i) = \sqrt{([\theta_1^d, \theta_2^d] - [\theta_1^i, \theta_2^i])\Sigma^{-1}([\theta_1^d, \theta_2^d] - [\theta_1^i, \theta_2^i])^T} \quad (4)$$

where Σ is a covariance matrix corresponding to angular measurement error. In our formulation, we allow for a DSM 2DOC to have multiple image 2DOC matches. This reduces the possibility of missing correct matches since the shortest Mahalanobis distance might not necessarily indicate the correct match due to noise and intrinsic discrepancies between the two heterogenous data sources. This effect is more pronounced for repetitive structures such as Manhattan grid-structured urban environments with most of the buildings and their structural corners sharing similar orientations.

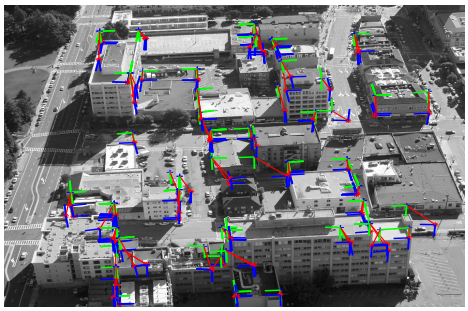


Figure 7. 264 putative matches after the Hough transform: blue intersections denote image 2DOCs, green intersections denote projected DSM 2DOCs, and red lines indicate the correspondence between them.

3.4. Hough transform based on rotation

A large number of, not necessarily correct, putative matches are typically generated from the previous step. For instance, there are 3750 putative matches between 2DOCs in Figures 5 and 6. It is necessary to obtain four correct inliers simultaneously in order to fit a Homography matrix resulting in a large number of required iterations in GMSAC. Specifically, let $p_{inliers}$ be the ratio of the inliers among all the putative matches, and p_{conf} be the desired confidence level; then the required number of iterations in GMSAC is $log(1-p_{conf})/log(1-p_{inliers}^4)$ [7]. With fewer than 150 correct matches found manually out of the 3750 in our sample image, the number of required iterations is nearly 3 million for 99% confidence level. We assume Homography because the camera position error which is less than 3 meters from the GPS device is sufficiently small compared to the distance between the camera and the buildings on an image which is larger than 300 meters. Thus, the difference between the projected DSM 2DOCs and the corresponding image 2DOCs can be considered to be purely due to the camera rotation. We apply Hough transform to identify the rotation with the maximum consensus among the putative matches within which significant number of outliers may exist. Matches that do not satisfy this rotation constraint are then eliminated resulting in significantly fewer putative matches and hence fewer required number of iterations for a specific level of confidence. For instance, this results in only 264 matches out of the 3750 matches on the sample image shown in Fig. 7.

3.5. GMSAC based correct matches identification

We use GMSAC, a combination of generalized RANSAC [19] and M-estimator Sample Consensus (MSAC) [18], in order to further prune 2D corner matches. We use Generalized RANSAC to accommodate matches between a DSM 2DOC and multiple image 2DOCs. MSAC is used for its soft decision, which updates according to the overall fitting cost and allows for continuous Homographic

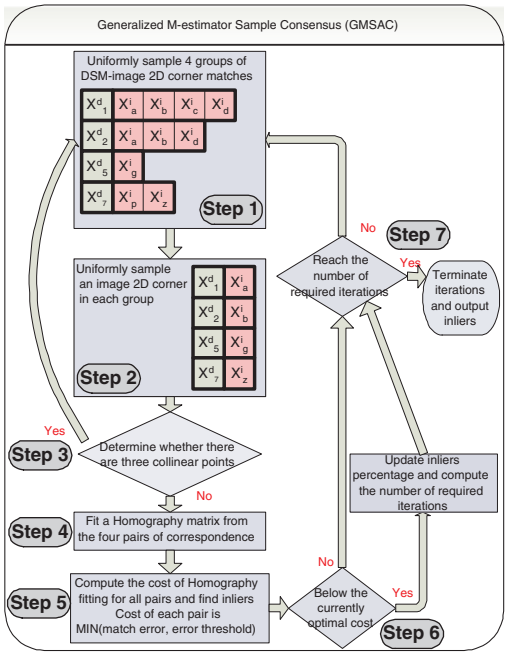


Figure 8. Block diagram for GMSAC algorithm.

model improvement. The details of GMSAC are presented as a block diagram in Fig.8 and can be describes as follows. 1) Uniformly sample four groups of DSM-image 2DOC matches. 2) Inside each group, uniformly sample an image 2DOC. 3) Determine whether there are three collinear points, a degenerative case for Homography fitting. If so, go to Step 1. Otherwise, move on to Step 4. 4) With four pairs of DSM-image 2DOC matches, a 3×3 Homography matrix, $\mathbf{H}$, is fitted with the least squared error [13]. 5) Every pair of DSM-image 2DOC match in every group is then examined with the computed Homography matrix from Step 4. Inliers are identified if their squared deviation distances are below a given error tolerance threshold. The cost associated with each match is the minimum of the squared deviation distance, and the error tolerance threshold. 6) If the sum of the costs is below the current minimum cost, $p_{inliers}$ is updated and the number of required iterations to achieve the desired confidence level is recomputed. Otherwise, another iteration is performed starting from Step 1. 7) Terminate the algorithm and output 2DOC match inliers if it has reached the required iteration number. Applying GMSAC to our sample image results in 134 matches as shown in Fig.9, which upon manual examination are verified to be correct. Due to the significantly higher $p_{inliers}$ obtained from Hough transform in the previous step, we obtain these 134 matches with fewer than 100 iterations in contrast with 3 million iterations needed without pruning via Hough transform.

Finally we apply Lowe's camera pose recovery algorithm [12] to all the identified corner correspondence pairs

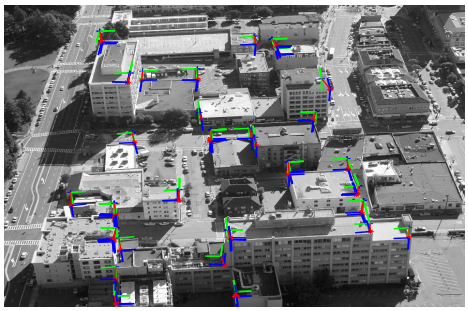


Figure 9. 134 correct DSM-image matches after GMSAC: blue intersections denote image 2DOCs, green intersections denote projected DSM 2DOCs, and red lines indicate the correspondence between them.

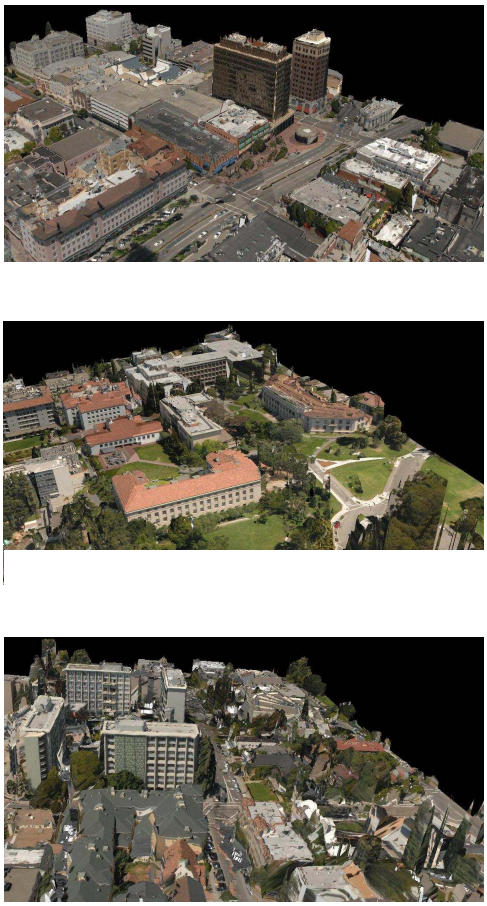
from GMSAC to obtain a more accurate set of camera parameters. Texture mapping from images to the 3D model is then performed according to [4].

4. Results

Our proposed system is tested with 358 aerial images taken during a 42-minutes helicopter flight over a 1.3km by 2.6km area in the city of Berkeley, California. The coverage area is divided into three regions with different characteristics. The first region is the downtown district, where large buildings are densely packed among few trees. The second region is Berkeley campus where large buildings are sparsely distributed among dense trees and vegetation. The rest of the area is grouped as residential where much smaller houses are densely packed among dense trees.

The correctness of the recovered camera pose is validated visually by examining the quality of fit between the projected DSM lines to the building contours in an image. When two sets of lines align sufficiently close to each other, the recovered pose is deemed to be correct for texture mapping. We have created several textured 3D models using recovered camera poses which have been visually validated. Fig.10 shows several screen shots of such 3D models for downtown, campus and residential models. The resulting textured 3D models can also be interactively viewed at http://www-video.eecs.berkeley.edu/ avz/aironly.htm. Each model corresponding to approximately $0.1km^2$ is textured with 9 images for the downtown and campus, and 8 images for the residential area. The texture alignment in Fig.10 shows that the poses rated as correct are indeed sufficiently accurate to result in visually pleasing texture mapped models. This visual evaluation has been further verified objectively by comparing the camera poses with the ones derived from manual correspondence [2].

In situations where the pose is visually confirmed not to be sufficiently accurate, we have investigated the source of error by examining the extracted 2DOCs from both the image and the DSM. The sources of error can be classified



(c)

Figure 10. Screen shots of the texture mapped models from aerial images with the camera poses estimated using the approach in this paper: (a)downtown; (b) campus; (c) residential area

into (a) too few 2DOCs matches and (b) dominant incorrect pose. Clearly, our system has no chance of finding the right camera pose when there are too few correct 2DOC matches. This happens when not enough true 2DOCs are extracted either from the image, or from the DSM, or both. By "true" 2DOCs, we mean those corresponding to actual intersections of two orthogonal building structure lines. Even with sufficient number of correct 2DOC matches, it is possible that some combination of 2DOC matches accidentally yield a random camera pose with a small fitting error. This is related to the ill conditioned nature of camera pose recovery problem. Both the Hough transform and GMSAC are based on the assumption that the pose with maximum consensus is the true camera pose. This assumption tends to break down when $p_{inliers}$ is significantly small.

Table 1 shows the correct camera pose recovery rate for our proposed approach, for downtown, campus and residential regions. Also shown are the associated reasons for incorrect pose estimates in each region. As seen, our algorithm achieves 91% correct recovery rate in the downtown region. In the remaining regions, it faces fundamental difficulties as suggested by the significantly lower recovery rates. By examining this regional performance difference, a major trade-off on the system performance is revealed. As stated earlier, one reason for camera pose recovery failure is that not enough true 2DOC matches are available. Even though it is possible to extract more 2DOCs by relaxing certain processing parameters such as minimum line length threshold, this could potentially lead to additional false 2DOCs, resulting in a decrease in $p_{inliers}$. This in turn would result in erroneous pose due to dominant incorrect 2DOC matches.

Region	Correct	Incorrect reason		% correct
		a	b	
Downtown	82	0	8	91
Campus	57	37	18	51
Residential	78	51	27	50
Total	217	88	53	61

Table 1. Correct pose recovery rate and sources of error in the proposed system for different regions. Incorrect reason a has to do with too few 2DOC matches; incorrect reason b has to do with dominant incorrect pose.

Let us now examine each of the three regions separately. In the downtown region, 2DOCs from both the DSM and the images are accurately extracted since the line extraction from the image and contour simplification from the DSM are both straightforward; furthermore, orthogonal structural corners are abundant due to both large and simple rigid building shapes. As such, there are plenty of correct 2DOC matches. In fact, erroneous camera pose for the 8 images in downtown is entirely due to too many incorrect 2DOC matches, leading to a low percentage of inliers, $p_{inliers}$.

The analysis on campus and residential areas reveals two fundamental difficulties in extracting true 2DOCs from a DSM. First has to do with the building density in a region. This effect is clearly observed by comparing the results from the downtown and campus areas. Both areas are characterized by large buildings; however, the building density is dramatically lower in campus than in downtown. For instance, some of the aerial images of the campus only contain a portion of a given building since the buildings tend to be larger than the ones in the downtown, and are further spread apart. Even if one or two buildings are present in an image, we might not be able to obtain enough 2DOCs due to imperfect contour simplification and complex building structures on the campus. Furthermore, this small number of 2DOC matches often does not provide enough constraint on Lowe's pose recovery algorithm. Since our algorithm requires a large number of 2DOCs for robustness and accuracy, it performs poorly in open fields where buildings are sparsely distributed.

Residential area has a similar building density to downtown. The performance however is much worse due to lack of true 2DOCs from the DSM. This is because the trees near houses are included as part of buildings after region segmentation, and the resulting region contours have many more sides than the ones from the buildings themselves without the trees. It is thus very difficult to extract true 2DOCs from these irregular shaped contours. At the same time, a large number of false 2DOCs have been included because of the trees. This effect is less severe in the campus model where the buildings are large enough that relatively small distortions on the contour due to tree occlusions can be removed by DP algorithm.

5. Conclusion and future direction

We have described an algorithm for registering oblique aerial imagery to 3D geometry model obtained from LiDAR, and demonstrated its performance over downtown, campus and residential areas of Berkeley using 358 images. Across all three regions, the proposed system achieves 61% correct camera pose recovery rate. In particular, its recovery rate in downtown district is 91%. At the same time, this system is computationally efficient. Specifically, the average processing time of 191 seconds per image with Intel Xeon 2.8GHz processor is significantly shorter as compared to over 20 hours of the exhaustive search per image reported in [4].

Future research should address the low building density and tree occlusion problems. Line matching between a DSM and images can potentially be simple and effective for the campus region since most of the long line segments are from buildings, and occlusion between buildings is insignificant. A parameterized roof region fitting could potentially be beneficial to both tree removal and 2DOC extraction in the residential area.

6. Acknowledgement

This research was supported by Army Research Office contract on Heterogeneous Sensor Webs for Automated Target Recognition and Tracking in Urban Terrain (W911NF-06-1-0076).

References

[1] S. Barnard. Interpreting perspective images. *Artificial Intelligence*, 21:435–462, 1983.

[2] M. Ding. Automated texture mapping in a 3d city model. Master's thesis, University of California at Berkeley, Dec. 2007.

[3] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10:112–122, oct 1973.

[4] C. Frueh, R. Sammon, and A. Zakhor. Automated texture mapping of 3d city models with oblique aerial imagery. In *In Proceedings of the 2nd International Symposium on 3D data processing, visualization and transmission*, pages 396–403, Sept. 2004.

[5] C. Fruh and A. Zakhor. Constructing 3d city models by merging aerial and ground views. *IEEE Comput. Graph. Appl.*, 23(6):52–61, Nov-Dec 2003.

[6] C. Fruh and A. Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60(1):5–24, Oct. 2004.

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[8] S. Hsu, S. Samarasekera, R. Kumar, and H. S. Sawhney. Pose estimation, model refinement and enhanced visualization using video. In *CVPR '00: Proceedings of the 2000 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 488–495, July 2000.

[9] J. Hu, S. You, and U. Neumann. Automatic pose recovery for high-quality textures generation. In *18th International Conference on Pattern Recognition*, pages 561–565, Aug. 2006.

[10] S. C. Lee, S. K. Jung, and R. Nevatia. Automatic integration of facade textures into 3d building models with a projective geometry based line clustering.

[11] L. Liu, I. Stamos, G. Yu, G. Wolberg, and S. Zokai. Multiview geometry for texture mapping 2d images onto 3d range data. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2293–2300, Washington, DC, USA, 2006. IEEE Computer Society.

[12] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1987.

[13] Y. Ma, S. Soatto, J. Kasecka, and S. S. Sastry. *An Invitation to 3-D Vision From Images to Geometric Models*. Springer, New York, 2004.

[14] U. Neumann, S. You, J. Hu, B. Jiang, and J. Lee. Augmented virtual environments (ave): Dynamic fusion of imagery and 3d models. In *VR '03: Proceedings of the IEEE Virtual Reality 2003*, page 61, Washington, DC, USA, 2003. IEEE Computer Society.

[15] J. A. Shufelt. Performance evaluation and analysis of monocular building extraction from aerial imagery. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(4):311–326, 1999.

[16] J. A. Shufelt. Performance evaluation and analysis of vanishing point detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(3):282–288, Mar. 1999.

[17] I. Stamos and P. K. Allen. Geometry and texture recovery of scenes of large scale. *Comput. Vis. Image Underst.*, 88(2):94–118, 2002.

[18] P. H. S. Torr and A. Zisserman. Mlesac: a new robust estimator with application to estimating image geometry. *Comput. Vis. Image Underst.*, 78(1):138–156, 2000.

[19] W. Zhang and J. Kosecka. Generalized ransac framework for relaxed correspondence problems. In *International Symposium on 3D Data Processing, Visualization and Transmission*, 2006.

[20] W. Zhao, D. Nister, and S. Hsu. Alignment of continuous video onto 3d point clouds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1305–1318, 2005.