An Automatic Method for Matching 2D ADS40 Images onto a 3D Surface Model

Zhen Liu[1,2], Peng Gong[3], Peijun Shi[4], Houwu Chen[2], T Sasagawa[5]

[1]College of Educational Information Technology, Beijing Normal University,100875, liuzhen@bnu.edu.cn
[2]Center of Information & Network Technology, Beijing Normal University,100875
[3]Department of Environmental Science, Policy, and Management, University of California, Berkeley, USA CA 94720-3114
[4]College of Resource Science & Technology, Beijing Normal University,100875
[5]Institute of GIS, Pasco Corp, Tokyo

Abstract

An automatic method is developed here to paste aerial images onto an urban 3D surface model for more realistic visualization. In this study we extracted different side views of urban constructions from an aerial image acquired with an airborne linear scanner sensor. We then matched those sideview images onto a 3D surface model according to the correspondence between the image and model. Side view feature extraction from images and matching those features to 3D models are two key steps in developing an automatic 3D image modelling technique. Here we present a new line-extraction approach using a multiple-level feature filter, which consists of the following: a Canny edge detector, an edge phase filter, an edge direction filter with fault tolerance, a Hough transformer, and a neighbouring line-segment fuser. We propose a base-line segmentation and parallelogram extraction algorithm based on perceptual organization. The algorithm employs uncertainty reasoning and is based on part forms for shape expression. It is computationally less intensive and noise free. Matching 2D images to 3D models requires finding a transformation matrix to minimize error. A lot of algorithms have been presented to solve the matching problem. However, there is still no good solution to the problem as it has too many unknown parameters. In this research, we first project images based on the camera model after a partial matching between the extracted parallelogram and the 3D model is carried out. Then, the Hausdorff distance is calculated between edges in the original image and the projected image, based on which sideview feature mapping is realized to obtain 3D virtual views based on a 3D surface model and a 2D image.

Keywords

3D model, ADS40 image, Feature extraction, Feature matching

I. INTRODUCTION

Virtual reality based on 3D buildings is successfully employed today for real-time visualization in such diverse fields as urban planning and architectural design. Visualization of 3D buildings is necessary for nearly all science and engineering disciplines, in order to easily assess disasters, object grouping, environmental goodness-of-fits, and many other applications. There is no doubt that 3D realistic view construction can be manually or semi-automatically done based on 3D reconstruction from stereo images based on photogrammetric principles (Sheng et al., 2001; Gong et al., 2002). Although a number of software packages offer methods for 3D model reconstruction, 2D image projection to 3D models, the processing is often time-consuming and circumstantial (Gruen et al., 2003). There are a few examples of automatic 3D image view construction. Previous research, especially from the University of Southern California and the University of Stuttgart, has found that because of noise, occlusion, and lack of camera information, extracting buildings from monocular aerial images in urban areas is not feasible (Dieter, 2003), while matching 2D images to 3D models is also difficult without knowing anything about the camera parameters.

In this paper, we propose an automatic method for projecting 2D images to 3D surface models with data acquired from an airborne linear scanner sensor and a 3D model of urban surface. The method involves extracting building sideview features from images, matching extracted features to the 3D model, and mapping image features according to the correspondence between the image and model. In the following we present the algorithm in detail.

II. METHOD

A. Edge extraction

Edge detector

The Canny edge detector has been shown to be optimal for images corrupted by Gaussian white noise (Canny, 1986), and is used to detect edges in the present paper. The detector is demonstrably more effective than the LoG operator, Sobel operator or other operators. Using the Canny Edge detector, we obtain not only edge intensity information, but also the edge phase (direction), which is used in later steps.

The Canny operator works in a multi-stage process. First, the image is smoothed by a Gaussian convolution. Second a simple 2-D first derivative operator (somewhat like the Roberts Cross) is applied to the smoothed image to highlight regions of the image with high first derivatives. Edges give rise to ridges in the gradient magnitude image. Third, the algorithm tracks along the top of these ridges and sets to zero all pixels that are not

1082-4006/06/1202-92$5.00

actually on the ridge top so as to give a thin line in the output, a process known as non-maximal suppression. Finally, the phase of the edge is calculated by taking the ratio between the first two derivatives(http://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm).

Phase filter: It is reasonable to assume that the neighbouring pixels of straight contours from man-made object would have a similar phase. Thus, we designed the algorithm as described below.

Labeling the phase of the edge pixels: For each pixel, label the surrounding pixels from 0 to 7, and assign its phase. We do not need to calculate an inverse trigonometric function for each edge pixel; instead, we only perform additions and multiplications on the derivative in the x-axis, the derivative in y-axis, and the invert of tan(PI/8), to reduce computation.

Tracking the edge: For each pixel *P*, we check its 8 neighbouring pixels. If a neighbouring pixel *Pn* has a phase similar to *P*, then we continue tracking *Pn*. If this tracking results in too few pixels, according to a preset threshold value, we can reasonably assume that the tracked edge is noise and delete it.

Edge filtering according to direction: The described filtering process has no effect on curve edges whose edge phases do not change abruptly. Therefore, further filtering is necessary. For linear edges, connections between neighbouring pixels are typically in the same direction. Figure 1 shows that neighbouring pixels that have at most 8 connection directions produce a reciprocal oscillation phenomenon.

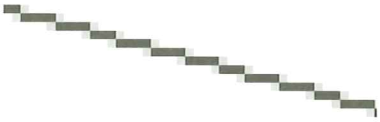


Figure 1. Oscillating direction

As a line is tracked, a fault tolerance in direction is applied as shown in Table 1.

Table 1. Line direction tracking

0	1	2
3		3
2	1	0

1) For edge pixel *P*, we track in a given direction. For each neighbouring pixel located in the direction *D* of *P*, if direction *D* has not been tracked, then we continue tracking. We set the fault direction counter *C* to be 0, and add the temporary tracking result to the final tracking result. Otherwise, we increase the direction fault counter *C* by 1 when tracking edge pixels located in direction *D* and will place the tracking result in a temporary buffer. We then record the direction code.
2) If counter C is greater than a threshold (set empirically), then we abandon the tracking result in the buffer.
3) If the final tracking result has too few pixels, then the edge is considered to be noise, which does not support line feature extraction.

This method is similar to some chain-code based line-detection techniques (Zhu, 2003; Bao et al., 2003; Sun et al., 2003; Shi et al., 1999), but it is more efficient and more effective in reducing noise.

B. Line segment extraction

Line segment extraction is achieved by line combination after Hough transformation. For each filtered line-support edge discovered by the abovementioned tracking process, a Hough Transformation (*HT*) is applied, resulting in a line segment. *HT* is a stable and robust line detection method (Luc, 1998), but its primary disadvantage is its large computational overhead. In this paper, *HT* is modified to reduce computational overhead. We describe our method below.

1) We calculate the centre of each line-supporting tracking result. We then set a new coordinate system with the centre as its origin, and express the tracking result in this new system. Thus, the radius is reduced to study only a small area around the centre.
2) We estimate the range of possible angles according to the retained direction code.
3) We apply HT, select the parameter with a maximum vote in the HT space, and finally, select the pixels that vote for it. We then determine the two end points of the extracted line according to the maximum and minimum coordinates among the selected pixels.
4) We express the line in the original coordinate system.

This method avoids several disadvantages of *HT*, such as, rapid performance reduction from larger image size, difficulties in determining the voting threshold when there are multiple lines to be extracted, and difficulties in determining the end points of an extracted line. However, the generality of HT is lost in this process, so further processing is necessary. For each extracted straight line, we search the neighbouring areas of its two end points.

If another line is found, and if the two lines proceed in a similar direction, then we combine the two lines. For each line in a line-set, if another line is found that can be combined in the same set, the line-set can be combined. Through this process, short lines, such as lines *a* and *b* in Figure 2, are combined into one long line, while many noisy lines, such as line *c* in Figure 2, are removed.

C. Parallelogram extraction from building in images

To extract parallelograms, the theory of perception organization and uncertainty reasoning are employed. Perception organization, first proposed by Lowe (David, 1985) in his SCERPO system, has been recognized by many researchers

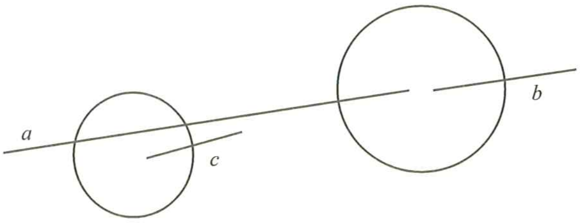


Figure 2. Line combination

for its ability to derive global structures from local primitives. The theory is applied in numerous projects due to its low computational requirement and high anti-noise capabilities (Lin, 1996; Williams and Hanson, 1996; Christopher et al., 1994). Based on the characteristics of parallelograms, the following algorithm, shown in Figure 3, is used to adapt perception organization to extract parallelograms from extracted lines.

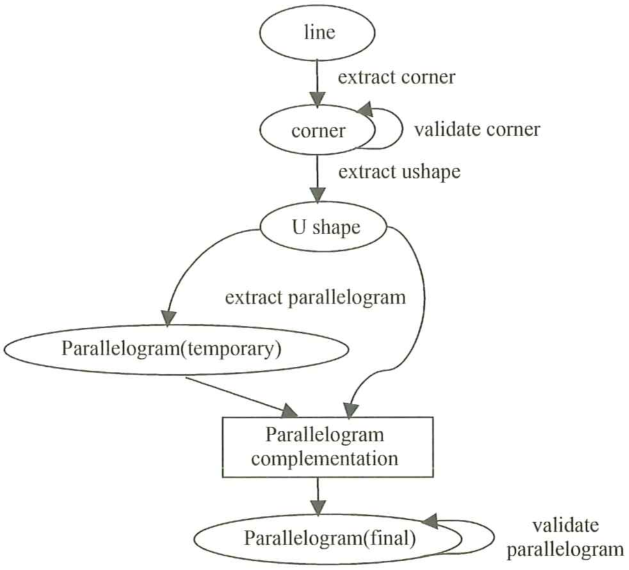


Figure 3. Flowchart for parallelogram extraction

Uncertainty reasoning is also applied to solve the following problem. Although Bayesian reasoning and Dampster-Shafer's evidential theory are used by others (Vasseur and Pegard, 1999; Mei, 1997; Xi, 2000), a simpler but effective method is used in this paper. The combination of multiple evidence is described below:

1) Bel(A1 and A2 and … An)=min{Bel(A1),Bel(A2), … Bel (An)};
2) Bel(A1 or A2 or … An)=max{Bel(A1),Bel(A2), … Bel(An)};
Bel(X) denotes belief for X;

(i) Shape representation
All shapes, including corners, U-shapes and parallelograms are represented by parts defined as follows.

Definition: For a point set P in a 2D plane**,** the two end points of line segment a is *point1* and *point2,* while the coordinates of an arbitrary point in line a is *(x,y).* If |*point1.x-point2.x*| <|*point1.y-point2.y*|, then point set $\{(p,y) \mid p < x, (p,y) \in P\}$ is called the left side of line a, $\{(p,y) \mid p > x, (p,y) \in P\}$ is called the right side of line a, and the two topples *(a, pos)* are called a *part, where pos* is a flag to denote left or right. In a similar way, we can define the case for |*point1.x-point2.x*|> |*point1.y-point2.y*|.

Thus corners, U shapes and parallelograms can be expressed by 2 parts, 3 parts, and 4 parts.

(ii) Corner extraction:
For a possible corner C1 shown in Figure 4, the belief of a corner can be calculated by the following

$$Bel_{a1} = 1 - dist(P,P) / dist(P,P_{a1}) = 1$$

$$Bel_{a2} = \begin{cases} dist(P,P_{a_1}) > lenTh, & 1 \\ else, & 0 \end{cases}$$

$$Bel_{b1} = 1 - dist(P,P_{b1}) / dist(P,P_{b0})$$

$$Bel_{b2} = \begin{cases} dist(P,P_{b0}) > lenTh, & 1 \\ else, & 0 \end{cases}$$

where $dist(p_1,p_2)$ denotes the distance between point p_1 and p_2, $Bel(X)$ is the belief that X and *length* is the minimum side length of an accepted corner. Then, the belief of $C1$ can be expressed as:

$$Bel(C1) = \min\{Bel_{a0}, Bel_{a1}, Bel_{b0}, Bel_{b1}\}$$

If a belief threshold is given, then any corner with a belief greater than the threshold is accepted.

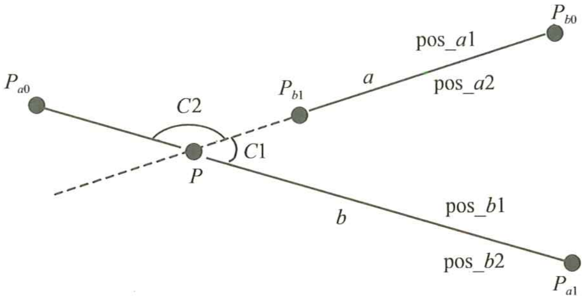


Figure 4. Corner extraction

(iii) U-shape and parallelogram extraction:
As in Figure 5, a corner structure *C1* consists of part1=(a, pos_a),part2=(b, pos_b), if another corner structure *C2* consisting of part3=(c, pos_c) and part2 exists. We can then compute the belief that the parallel structure P consisting of part3 and part1 are as follows:
If a is not located in the *pos_c* side of c, or c is not located in *pos_a* side of a then *Bel(p)=0;*
Otherwise,

$$Bel(P) = \begin{cases} 1 - angle(a,c) / Angle, angle(a,c) < Angle \\ 0, \quad else \end{cases}$$

where *angle(a,c)* denotes the inclination of a and c, and *Angle* is the maximum inclination between two lines in an accepted parallelogram structure.

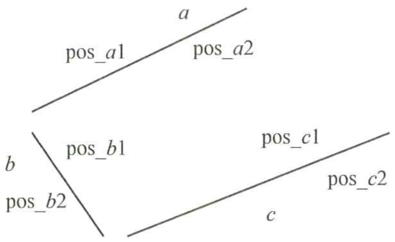


Figure 5. U-shape and parallelogram extraction

(iv) Parallelogram complementation

For an extracted U-shape, if no parallelogram is extracted from it, and if *Bel*(*U*) is greater than a given threshold, it could be assume that it is in fact the contour of a parallelogram; but, the fourth side had been lost because of line-extraction. The parallelogram can be closed with a fourth side *d*, as in Figure 6.

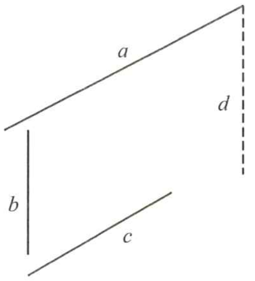


Figure 6. Parallelogram complementation

(v) Solving conflict

For the extracted parallelograms, space conflicts could appear. The following rules are employed to solve this problem.

1) Block off rule: As shown in Figure 7(a), both a,b and a.c can form a corner structure, but because b blocks a and c, the corner formed by a and c should be deleted.

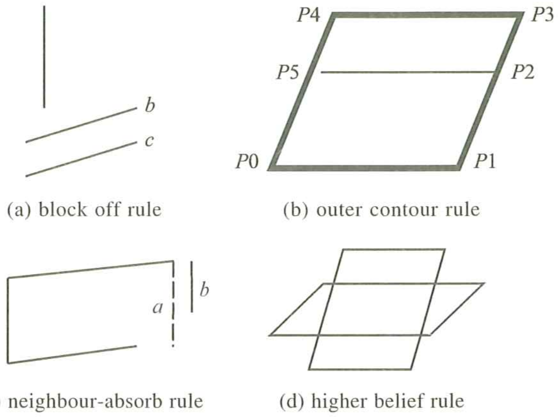
2) Outer contour rule: As shown in Figure 7(b), parallelogram (P0,P1,P2,P5) is contained in parallelogram (P0,P1,P3,P4), so the former is deleted.

3) Neighbour-absorb rule: As shown in Figure 7(c), side a is a line generated as a parallelogram complement, but it is near to b and has similar direction as b. Therefore, a should be absorbed by b, that is, we should delete a and enlarge b.

4) Higher belief rule: As shown in Figure 7(d), the parallelogram with lower belief should be deleted.

D. Matching between 2D images and 3D models

Matching between 2D images and 3D models is, in essence, the search for a transformation matrix that minimizes error. This is a very important and difficult task to achieve for mapping side images for buildings. In this paper, cursory matching will be done based on camera parameters before fine matching. Cursory matching can limit matching in a small range to reduce calculations, as well as matching errors.



(a) block off rule (b) outer contour rule

(c) neighbour-absorb rule (d) higher belief rule

Figure 7. Conflict rules

The transformation matrix for perspective projection can be written explicitly as a function of its five intrinsic parameters ($\alpha,\beta,u_0,v_0,\theta$) and its six extrinsic parameters (three angles defining rotation matrix *R* and three coordinates defining translation vector *t*)

$$M = \begin{pmatrix} \alpha r_1^T - \alpha \cot\theta\, r_2^T + u_0 r_3^T & \alpha t_x - \alpha \cot\theta\, t_y + u_0 t_z \\ \frac{\beta}{\sin\theta} r_2^T + v_0 r_3^T & \frac{\beta}{\sin\theta} t_y + v_0 t_z \\ r_3^T & t_z \end{pmatrix}$$

where r_1^T, r_2^T and r_3^T denote the three rows of the matrix *R*, and t_x, t_y, and t_z are the coordinates of vector *t*. There are 11 parameters to solve for. If the information for these parameters is not available, the parameter space is very large, making the computation unacceptably large. Various researchers have proposed algorithms to solve the matching problem, but there exists no effective solution, as there are many unknown parameters (David and Ronen, 1997; Ronen and Daphna, 1996; Steven et al., 1998). Our algorithm will simplify the camera model according to the image and CAD data given. For example, we make the following assumptions: $\alpha=\beta$, $\theta=0$, $R=I$ (*I* is a unit matrix), and setting u_0, v_0 as the centre of the aerial image. Thus, unknown parameters are fewer, and the difficulty of the matching problem will decrease, making the algorithm practical. Based on the camera model parameters obtained for our data set, for an arbitrary part of the aerial image, only u_0 and v_0 are unknown. The algorithm can move the aerial image along the model-projecting image, and, for each position (u_0, v_0), calculate the Hausdorff distance. The position that has the minimum Hausdorff distance is the optimal matching result. Hausdorff distance can be defined between point sets, line segment sets, and parallelogram sets with lower computational overhead. Yet, the reliability of the process also decreases. The following flowchart, shown in Figure 8, is employed to finish the matching between features in the image and the 3D model.

E. Side view image mapping

Since the camera positions of the original photographs are recovered during the modeling phase, projecting the images onto the 3D model is straight-forward. The process of mapping the side views of buildings from a single image onto the 3D model can be thought of as replacing each camera with a slide

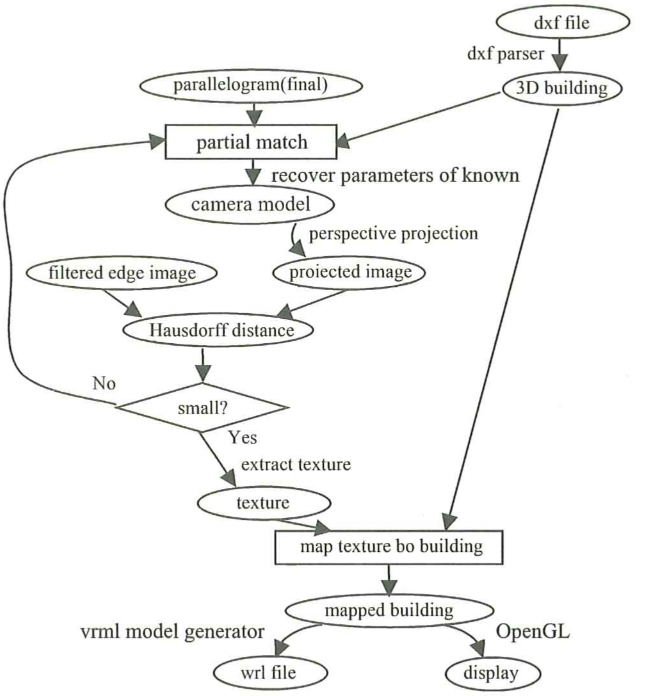
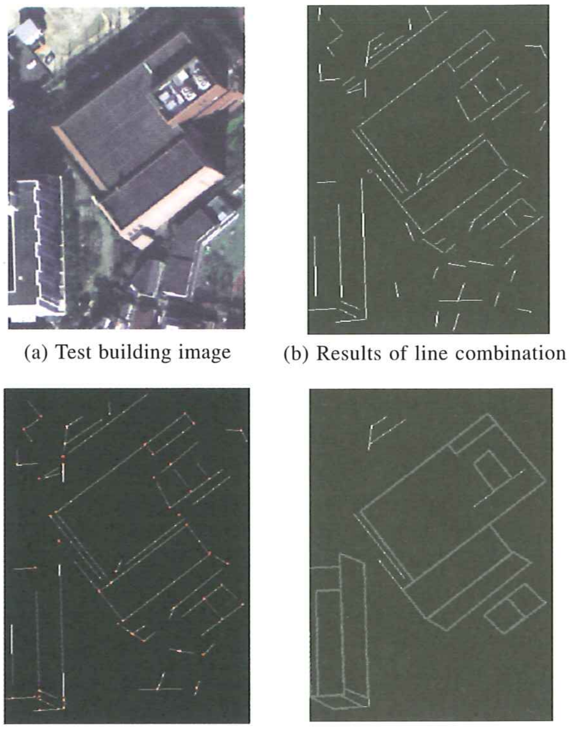


Figure 8. Matching flowchart


(a) Test building image (b) Results of line combination

(c) Result of corner extraction (d) Result of parallelogram

Figure 9. Building features extraction

projector that projects the original image onto the model. When the model is not convex, it is possible that some parts of the model will shadow others with respect to the camera. Such shadowed regions could be determined using an object-space visible surface algorithm, or an image-space ray casting algorithm, which is efficiently implemented using z-buffer hardware.

III. RESULT

We have two data sets for experiment. An aerial image and a 3D model in CAD. The 4 band multispectral image used in this study was acquired by an ADS40 SP1 linear sensor with a spatial resolution of 0.2 m. The 4 bands include NIR, R, G and B channels. The data covered the downtown of Shinjuku, Tokyo. The second data set used is a 3D model in a CAD file (DXF) of the same region. Figure 9 is an illustration of the building feature extraction algorithm. Figure 9(a) is taken from the image. There are many false edges after edge filtering and Hough transformation initially. After line combinations, most residual edges are outlines of buildings (Figure 9(b)). Figure 9 (c) shows the results of corner extraction. From Figure 9(d), some parallelograms of the un-shadowed sides of buildings are extracted, while some of them are not resumed. Based on some of the parallelograms, partial matching can be done.

After matching the 2D image to the 3D model, 3D sideviews of buildings can be achieved (Figure 10). Our results suggest that the algorithms of 3D sideview mapping for buildings based on matching the features in 2D images to 3D models is reliable and efficient. However, there are still some errors in the final results including some incorrect positions from the image are being mapped to the 3D surface model. By analyzing the results, errors are found to be derived from the following factors: 1) errors from the given CAD model; 2) errors caused by the ADS40 sensor in flying. 3) errors from calculation.

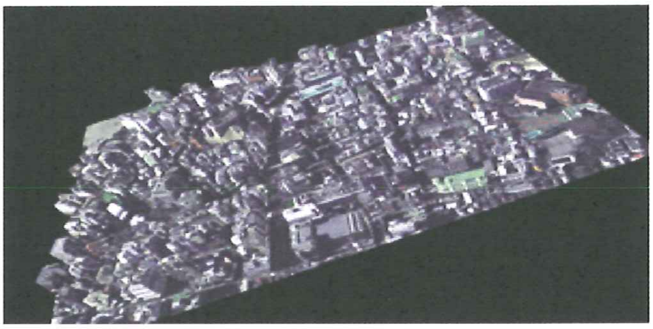


Figure 10. Results of projected side views of buildings

IV. CONCLUSION

The algorithm for parallelogram extraction based on perceptual organization theory is effective in resolving the problem of feature extraction from buildings using high spatial resolution images that have more details and noise. The relation between coordinates of the image and the coordinates of the 3D model can be integrated after matching the 2D image to the 3D model. The results suggest that an automatic sideview maping onto

3D buildings, as proposed here in this paper, can be achieved. If more accurate parameters can be given for the camera model, the final results can be more exact. Further studies must be done to improve accuracies of the feature extraction algorithm, and to make the matching more robust to noise perhaps by using artificial intelligence.

ACKNOWLEDGEMENT

This research is partially supported by a hundredth talent grant to Gong.

REFERENCES

[1] Bao S., Yang L., 2003, Algorithm for Detecting Vertical and Horizontal Lines in Real-time Image processing. *Journal of Hefei University of Technology*, 26(4): 550–552.

[2] Canny J F., 1986, A computational approach to edge detection. IEEE Trans. *Pattern Analysis and Machine Intelligence*, 8(6): 679–698.

[3] Christopher O. Jaynes, Frank Stolle, Collins R T., 1994, Task Driven Perceptual Organization for Extraction of Rooftop Polygons. Applications of Computer Vision, *Proceedings of the Second IEEE Workshop on 1994*.

[4] David G. Lowe, Perceptual Organization, Visual Recognition, 1985, *Kluwer Academic Publishers*, Norwell, MA.

[5] David Jacobs, Ronen Basri. 3-D to 2-D Recognition with Regions. *1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*

[6] Dieter F., 2003, 3D Building Visualisation-Outdoor and Indoor Applications, in Fritsch (Ed.). *Photogrammetric Week '03*. Wichmann Verlag, Heidelberg, pp. 281–290.

[7] Gong P., Y. Sheng, G S. Biging, 2002, 3D model-based tree measurement from high resolution aerial imagery, PE&RS, 68 (11): 1203–1212.

[8] Gruen A., Zhang L., Wang X., 2003, 3D City Modeling with TLS (Three Line Scanner) Data. International Archives of the Photogrammetry, *Remote Sensing and Spatial Information Sciences*, Vol. XXXIV-5/W10, 24–27 February, Scuol-Tarasp, Switzerland.

[9] Lin C., 1996, *Perception of 3-D Objects From an Intensity Image Using Simple Geometric Models*. PhD thesis, Faculty of the Graduate School, University of Southern California, pp. 28.

[10] Luc Baron, P. Eng., 1998, Genetic Algorithm for Line Extraction. Rapport technique EPM/RT-98/06, École Polytechnique de Montréal, pp. 20.

[11] Mei X., 1997, *Automatic 3D Modeling of Regular Houses from Aerial Imagery*. PhD Thesis. Wuhan Technical University of Surveying and Mapping.

[12] Ronen Basri, Daphna Weinshall, 1996, Distance Meteric between 3D Models and 2D Images for Recognition and Classification. *IEEE Trans. On Pattern Analysis and Machine Intelligence*, 18 (4): 465–470.

[13] Sheng Y., P Gong, G S. Biging, 2001, Model-based conifer crown surface reconstruction from high-resolution aerial images, PE&RS, 67(8): 957–965.

[14] Shi C., Xu S., Jing R., et c., 1999, The Replace Algorithm of Hough Transform in Real-time Image Processing. *Journal of Zhejiang University (Engineering Science)*, 33(5): 482–486.

[15] Steven Gold, Anand Rangarajan, Chien-Ping Lu, et c., 1998, New Algorithms for 2D and 3D Matching: Pose Estimation and Correspondence. *Pattern Recognition,* 31(8): 1019–1031.

[16] Sun A., Tan Y., 2003, Line Detection Algorithm Of Aerial Image. *Journal of Wuhan University of Technology (Transportation Science & Engineering)*, 27(6): 807–809.

[17] Vasseur P., Pegard C., 1999, Perceptual Organization Approach Based on Dempster-Shafer Theory. *Pattern Recognition,*32(8): 1449–1462.

[18] Williams L R., Hanson A R., 1996, Perceptual completion of occluded surfaces. *Computer Vision and Image Understanding*, 64(1): 1–20.

[19] Xi X., 2000, *Research on Model Based 3-D Object recognition in remote sensing image*. Dr. thesis. National University of Defense Technology.

[20] Zhu X., 2003, Extract Straight Line From Image Based on Computer Vision Task. *Journal of Nanchang Institute of Aeronautical Technology (Natural Science)*, 17(3): 63–66.