Journal of information and communication convergence engineering
http://jicce.org

J. Inf. Commun. Converg. Eng. 10(4): 411-419, Dec. 2012 **Regular Paper**

City-Scale Modeling for Street Navigation

Fay Huang[1*] and Reinhard Klette[2], *Member, KIICE*

[1]Institute of Computer Science and Information Engineering, National Ilan University, Yi-Lan 26047, Taiwan
[2]Department of Computer Science, The University of Auckland, Auckland 1142, New Zealand

Abstract

This paper proposes a semi-automatic image-based approach for 3-dimensional (3D) modeling of buildings along streets. Image-based urban 3D modeling techniques are typically based on the use of aerial and ground-level images. The aerial image of the relevant area is extracted from publically available sources in Google Maps by stitching together different patches of the map. Panoramic images are common for ground-level recording because they have advantages for 3D modeling. A panoramic video recorder is used in the proposed approach for recording sequences of ground-level spherical panoramic images. The proposed approach has two advantages. First, detected camera trajectories are more accurate and stable (compared to methods using multi-view planar images only) due to the use of spherical panoramic images. Second, we extract the texture of a facade of a building from a single panoramic image. Thus, there is no need to deal with color blending problems that typically occur when using overlapping textures.

Index Terms: City models, Image-based modeling, Panoramic imaging, Street models

I. INTRODUCTION

Various consumer electronic applications require 3-dimensional (3D) models of street scenes, for example for virtually flying or driving through a city, augmented reality, urban planning, or for documentation purposes (of city scenes at a particular point in time). The design of fully or semiautomatic methods for time-efficient modeling of city scenes has become has become an active research area, with manifold interesting contributions already in computer graphics and computer vision.

Due to the rapid increase of digital photography, image-based modeling approaches are today a cost-efficient and feasible way for doing large-scale city modeling. Often it is actually not required to provide very detailed reconstructions of buildings; for applications such as virtual tours or path planning, it is sufficient to provide some kind of 3D sketches. Even only 'rough' 3D textured street views will define progress compared to common GPS-based 2D maps or textureless road models, as typically used today in cars as a navigation tool. A visual impression of the 3D environment helps to navigate in unknown areas. This can be achieved by supplying realistic 3D street views of a route, for example, by simplified but realistic looking, texture-mapped 3D building models.

The aim of this paper is to develop a framework that takes a set of dense spherical panoramic images and an orthogonal aerial image of that area as input image data, and outputs texture maps and height information of the buildings through a semi-automatic process. The reconstructed 3D building models are textured with building elevation images, automatically extracted from the panoramic images, for

Received 18 September 2012, **Revised** 14 November 2012, **Accepted** 28 November 2012
***Corresponding Author** E-mail: fay@niu.edu.tw

Open Access **http://dx.doi.org/10.6109/jicce.2012.10.4.411** print ISSN: 2234-5973 online ISSN: 2234-8883

providing a realistic impression of the street view, as illustrated in Fig. 1.

II. RELATED WORK

Progress in sensor technology (e.g., laser scanners or panoramic cameras) has contributed in recent years to the success of 3D modeling techniques by providing various options for input data [1]. Aerial photogrammetry provides height maps (in particular the height of buildings) using stereo images or laser scanners [2-5]. Those height maps support the construction of 3D city models. However, models derived from aerial scans typically lack realism at ground level because aerial image data carry only limited information about textured side views (i.e., the facades) of buildings.

Automated texture mapping for 3D city models has been studied in [6-11] using ground-level image data. There are two major issues, namely visible seams in calculated textures due to variations in lighting or image resolution, and excessive time requirements due to the need for automated pose recovery for the applied multi-view approach (using planar ground-level images).

Some authors have suggested using recorded images only for reconstructing 3D street models [12-14], without including any depth information recorded by a laser scanner. For example, [13] discusses a plane-sweep algorithm where depth is estimated from multiple images. This ground-level image-based approach generates 2.5D models only rather than 3D models.

The authors of [15] propose the use of cylindrical panoramic images (for texture mapping based on ground-level data). They assume an approximate localization of recorded panoramas, and apply a voting process for the process of registration of aerial and panoramic images. They use sparsely recorded panoramic images, resulting in a lack of texture, and also a continuation of the problem of visible seams between overlapping textures.

Fig. 1. A street-view example showing 3-dimensional building models reconstructed following the proposed approach.

Fig. 2. A captured spherical panoramic image (left) and panoramic video camera used for image acquisition (right).

This paper proposes an image-based approach. We derive 3D building models from aerial images as publically available on Google Maps, and from recorded spherical panoramas. The generated models are approximate, represented by textured planar faces. Textures are extracted from spherical panoramas. The recovered camera trajectory defines essential information, and we describe how to achieve that by using global positioning system (GPS) data and selected key frames of spherical panoramas.

III. SYSTEM OVERVIEW

The input data of the developed system are from two types of image sources. They provide a dense set of ground-level spherical panoramic images and a nearly-orthogonal aerial view. We also applied the GPS information associated with the recorded panoramic images. For panoramic image acquisition we used a Ladybug 3 camera (Point Grey Research, Richmond, Canada), mounted on the top of a car. This panoramic video camera consists of six high quality Sony charge-coupled device (CCD) sensors, which capture six planar images, five looking horizontally outwards and one looking upward, with a frame rate of 15 images per second. Six images were then stitched together using the multiperspective plane sweep approach of [16] to generate a 720-degree spherical panoramic image with image resolution 2,048 (width) × 1,024 (height) pixels.

An example of a captured panoramic image, which has been transformed into a rectangular representation, and the Ladybug 3 camera, are shown in Fig. 2. The car was also equipped with a GPS device. This location information is used for initialization and bundle adjustment in the camera path recovery process. The other input image is a nearly-orthogonal aerial image of the area to be reconstructed. The image is generated by stitching together different sections of local aerial image blocks, as available in Google Maps.

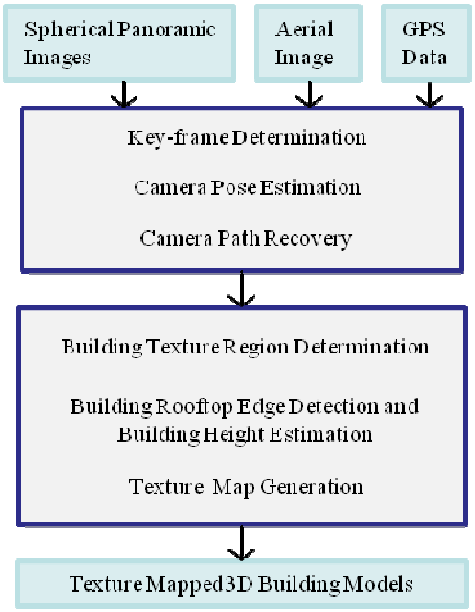


Fig. 3. Sketch of the proposed system pipeline.

For a sketch of the proposed system pipeline, see Fig. 3. The entire modeling process is grouped into two stages. The main task in the first stage is to recover the camera path based on video and GPS data.

The goal of the second stage is to generate building texture maps from available panoramic images with the help of specified clues (i.e., buildings' footprints) on the aerial image. Some reasonable assumptions about the input data of the proposed approach are described as follows:

The image projection model of each recorded panoramic image is defined by spherical projection with respect to a single projection center (i.e., we use a single-center projection model). A 3D Cartesian coordinate system is defined for each panoramic image, having its origin at its corresponding projection center. Each individual coordinate system is used for representing the location and orientation of the camera in the 3D space where a particular panoramic image is captured with respect to the world coordinate system. Moreover, the panoramic images are assumed to be recorded along a smooth and continuous path. GPS information is given defining some of the panoramic image acquisition locations. Furthermore, the aerial image as obtained from Google Maps is assumed to be an orthogonal aerial image, and building footprints on such an image are assumed to be given, using the same method as proposed in [15]. Building footprints are represented by a set of polygons, which can be calculated by preprocessing and segmentation of the aerial map, or simply by manual specification. The output of the system is a set of 3D polygonal building models, texture mapped by building textures extracted from the panoramic images. Fig. 1 illustrates a street-view of resulting 3D building models produced by the proposed approach.

IV. CAMERA TRAJECTORY ESTIMATION

The first stage in the system pipeline deals with the camera trajectory estimation task, which is an essential step towards accurate registration of two types of input images. Currently, registering the recovered camera trajectory to the aerial image is done manually by region-wise assignments of the camera path to the aerial map. Consequently, we derive the coordinates of images representing the locations in the aerial image associated with the recorded panoramic images.

Only selected key frames among a huge set of input video data (i.e., spherical panoramic images) are used to estimate the movement of the camera along the streets. A feature detection algorithm is applied to each of these selected panoramic images, and feature point matching is performed between successive selected images. Matching results enable us to recover the essential matrix describing the spatial relationship between both imaging coordinate systems. Relative orientation, represented by a rotation matrix, and position, represented by a unit vector, of both successive panoramic images are derived (as common in computer vision) from the essential matrix.

The camera trajectory is recovered based on a point cloud reconstruction of the scene and bundle adjustment based on available GPS information. The camera path is refined by registering it to the aerial image. The following subsections describe the methods used in each step.

A. Key-Frame Determination

The image acquisition rate of the Ladybug 3 panoramic video camera is 15 frames per second (fps). The average speed of the image acquisition vehicle is about 35 km/hr (of course not constant due to traffic-related events). Therefore, a large set of panoramic images is captured within a very short period of driving. Normally, one recording session is about 10 to 20 minutes. For the goal of recovering the camera's trajectory, a large fraction of the input image data may be considered to be redundant, because it is not only unnecessary for improving the accuracy of the estimated trajectory, but may even add more noise to the trajectory calculation process. Therefore, only a selected set of key frames is used for the camera trajectory recovery process. In other words, the task for the first step is to eliminate redundant images.

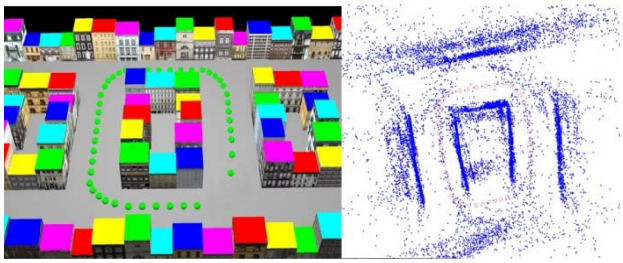
Algorithm 1 : Eliminating the Redundant Images

Input: Image frames P_i for $i = 1, 2, \ldots, m$.
Output: Image frames Q_i for $i = 1, 2, \ldots, n$. $(n < m)$
$j = 1$.
for $i = 1$ to m **do**
  $P_i = GaussianSmooth(P_i)$. % kernel size 9x9
  $B_i = SobelVertical(P_i)$. % B_i is a binary image
  $B_i = Dilation(B_i)$. % kernel size 9x9
  if $(i > 1)$ **then**
    $D_i = Difference(B_i, B_{i-1})$.
    $w_{i-1} = NumWPix(D_i) / NumWPix(B_{i-1})$.
    if $(w_{i-1} > s)$ **then** % s is a threshold
      $Q_j = P_i$.
      $j = j + 1$.
    end
  end
end

The first filter applied to the input sequence is to select only every *k*-th frame of the video sequence, where *k* can be determined by the actual average speed of the vehicle. From our experiments, a reasonable value of *k* is between 3 and 7. The second filter aims at eliminating image subsequences captured while the vehicle is stopped (e.g., at a red light). Algorithm 1 (see above) is used for this task. The methods involved in this algorithm include a *Gaussian* smoothing operator, the *Sobel* edge detector, and a dilation method. Particularly, only vertical edges were taken into account. The function *Difference* performs a logical exclusive OR on two binary images, B_i and B_{i-1}. Function *NumWPix* reports numbers of white pixels (i.e., intensity value equals to 1) in a binary image. Moreover, *s* is a threshold for this filtering task. In the conducted experiments, the value of *s* was set to be 0.1.

B. Camera Pose Estimation

For solving this task, we prefer the spherical representation of panoramic images rather than cylindrical images. This is because pairwise relative camera poses can be recovered by directly adopting well-known camera self-calibration methods (such as the 8-point algorithm) originally developed for planar images. This, in turn, is because a single projection centre has been assumed in the spherical image representation. Those algorithms have shown to be robust to error, and it is possible to achieve real-time performance. Note that, in order to adopt those camera self-calibration algorithms to the panoramic image situation, a single-centre projection constraint must be ensured.



(a) (b)

Fig. 4. (a) a synthesized 3-dimensional world for experiments; dots indicate the camera trajectory and (b) Recovered camera path and reconstructed scene points.

The camera pose estimation approach is as follows: First, a feature detection algorithm is applied to each of the selected panoramic images, and then a feature point matching search is performed between each pair of successive images. The matching results enable us to recover the essential matrix that describes the spatial relationship (i.e., camera pose) between two imaging coordinate systems. The spatial relationship between two successive panoramic images can be characterized by a rotation matrix, describing the relative orientation, and a unit vector, describing the relative location.

Correspondences between image points are established by scale-invariant feature transform (SIFT) feature detection plus SIFT-based matching [17]. In the algorithm, a single threshold, denoted as *t*, is used to determine whether a match is acceptable. The smaller the value of *t*, the more image correspondences are identified, and the higher is the possibility that the result would include false matches.

The 8-point algorithm is employed to estimate the essential matrix. The algorithm utilizes the epipolar constraint, more specifically the coplanarity constraint, for calculating the essential matrix. The coplanarity constraint can be assured by vector arithmetic; thus, the implementation of the 8-point algorithm is independent of the geometrical form of the image manifold (i.e., planar, spherical, or cylindrical). A 2-pass approach is proposed to obtain the final essential matrix. First, an initial essential matrix is derived according to a smaller set of corresponding image points, which is the matching result associated with a relatively large threshold value *t*. Those sparse corresponding points are believed to be more accurate but less descriptive. Next, a smaller threshold value *t* is assigned to obtain a larger set of point matches. The initial essential matrix is then used to serve as a constraint to filter out incorrect matches. In other words, the matching outliers are filtered out based on the epipolar constraint. Remaining point matches are then used to compute the final essential matrix.

Algorithm 2 : Camera Path Recovery

Input: R_i and T_i for $i = 1, 2, \ldots, n$. (camera poses)
M_i for $i = 1, 2, \ldots, n$-1. (point matching info. between Q_i and Q_{i+1})
Output: S (a set of scene points)
$C = \{C_i \mid i = 1, 2, \ldots, n.\}$ (a set of camera locations)

```
Initializing C1 and C2.
C = {C1 , C2 }.
S = CalculateScenePt(R2 , T2 , M1).
for i = 3 to n do
  S = UpdateScenePt_1(M_{i-1} , S).
  Ci = CalculateCamPos(Ri , Ti , S, C).
  S = UpdateScenePt_2(S, Ci).
  C = Insert(C, Ci).
  if i is a multiple of h, then      % for each h-th of Ci
     C = UpdateCamPos(C, GPSi ).
     S = UpdateScenePt_3(S, C).
  end
end
```

The derived essential matrix is used to solve for the external camera parameters R and T, which stand for the rotation matrix and the translation vector, respectively. The algorithm leads to two valid solutions for R and two solutions for T pointing in opposite directions. The correct solution for translation T can be obtained by assuming forward motion of the camera. To identify the correct solution for R, scene points based on already processed panoramic images are reconstructed with respect to the 3D world coordinate system. Each valid solution of R is used to calculate new scene points. Ideally, the majority of those new scene points should coincide with previously reconstructed scene points. The correct solution of R is then determined by evaluating these 3D reconstruction results.

C. Camera Path Recovery

Pairwise external camera parameters are integrated, one by one, to obtain global camera motion, and thus the camera's trajectory. The major drawback of such a method is that the camera parameter estimation error would propagate through the integration process. In order to deliver a more accurate and more efficient solution to this problem, the GPS information was incorporated into the path recovery algorithm, as summarized below in Algorithm 2. Since the accuracy of GPS varies from 1 to 5 m, it is sensible to correct the path drift every 50 m based on the GPS reading. The value of *h* in the algorithm is decided based on this criterion.

There are three different functions for updating the reconstructed 3D scene points in the algorithm. All of the scene points are stored in a database S, in which the visibility of each scene point with respect to each camera location is also available and is continuously updated by function *UpdateScenePt_1*. The purpose of function *UpdateScenePt_2* is to refine the scene point positions taken into account the newly recovered camera location. The third *UpdateScenePt* function is performed after each bundle adjustment takes place. It is also used to refine the scene point positions based on the newly updated camera path. Every *h*-th location of the estimated camera's path is shifted to coincide with the GPS reading. The amount of displacement is evenly distributed along the piecewise path. Fig. 4 illustrates a camera path recovery result.

V. GENERATION OF TEXTURE MAPS

The second stage in the system pipeline deals with the generation of texture maps for buildings based on the recorded panoramic images. At this stage, it is assumed that the camera path, which specifies the locations of key panoramic image frames, has been registered to the aerial image.

In order to extract building textures from the panoramic images, first, the image region containing the selected building is determined using the provided footprint boundaries of this building in the aerial image. Next, the system looks for the building's rooftop edge in the panoramic image within the estimated region. Once the building's rooftop edge is identified, it is able to extract more precisely a building texture map from this image region. Finally, the building texture must undergo a warping and rectification process before it can be used for texture mapping. The height of the building can also be estimated based on the image registration result and the location of the identified building's rooftop edge.

The usage of texture maps not only enhances the visualization of the 3D models but also verifies height information about the buildings. The following subsections report about techniques involved in the second stage of the system pipeline.

A. Determination of a Texture Region of a Building

For each building footprint, shown in the aerial image, our main interest is to extract the front facade view of the building from the panoramic image.

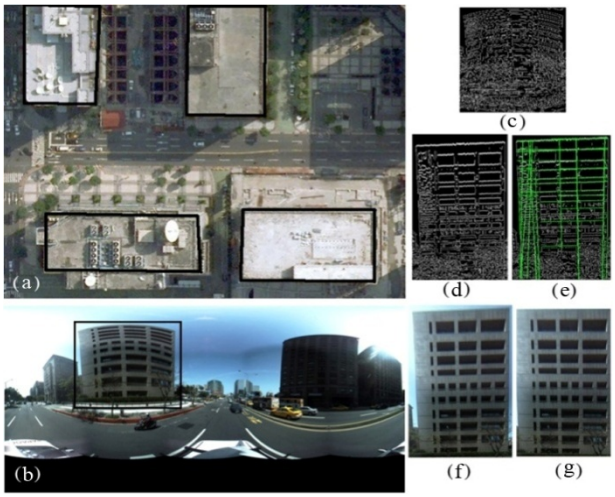


Fig. 5. (a) Section of an aerial image with buildings' footprints specified by black polygons (in this case a set of rectangles). (b) This panoramic image shows a rectangular region which contains the desired building texture. (c) This binary image represents the edge detection result of the rectangular image region in (b). (d) The back-projected image of (c). (e) The resultant image after straight line detection. (f) The back-projected image region in (b). And (g) Final extracted building texture after cropping and rectification.

An example of an extracted building texture and intermediate procedures are illustrated in Fig. 5. The first step is to determine a building's textured region, denoted by I_R, as shown by the bold black rectangle in Fig. 5b. A dense set of panoramic images was acquired, and the same building appears actually in several successive panoramic images. Thus, we are looking for a recorded panoramic image which contains the largest projected region of the desired building; this needs to be solved as a necessary initial task.

The system selects a panoramic image Q_i captured at the location denoted by aerial image coordinates (x_i, y_i), which minimizes the following distance measurement

$$D = \sqrt{(x_i - x_m)^2 + (y_i - y_m)^2}$$

where (x_m, y_m) represents the mid-point location of the line segment l, which is an edge of the polygonal footprint of the given building, and specifies the front facade.

Then, in the selected panoramic image Q_i, a rectangular region I_R, containing the desired building texture, is determined which is bounded by a top-left corner point at location (u_l, v_t), and a bottom-right corner point at location (u_r, v_b). The values u_l and u_r, where $u_l < u_r$ and $u_l, u_r \in \{1,2, \ldots ,W\}$, can be determined according to the width of the projection of l onto the panoramic image Q_i. Let u_s and u_e denote the projections of the start and end points of the footprint edge l, respectively. Moreover, W denotes the width of the panoramic image (in pixels) and H denotes the image height. We have

$$u_l = \min(u_s, u_e) - |u_s - u_e|/4, \text{ and}$$
$$u_r = \max(u_s, u_e) + |u_s - u_e|/4.$$

The vertical range of region l is specified by values of v_t and v_b, where $v_t < v_b$ and $v_t, v_b \in \{1,2, \ldots ,H\}$. These two boundary values can be obtained by the elevation of the camera location, denoted by h, the shortest distance between camera and the building, denoted by d, and the maximum possible height of the building, denoted by b. We have

$$v_b = H \times \left(\frac{\arctan(h/d)}{\pi} + \frac{1}{2} \right), \text{ and}$$
$$v_t = H \times \left(1 - \frac{\arctan(b-h)/d}{\pi} \right).$$

B. Rooftop Edge Detection and Height Estimation

Canny edge detection is now applied to region I_R (Fig. 5c). Then, the resultant binary image is back-projected onto a planar surface, denoted as I_P (Fig. 5d). Next, a Hough transform is employed to detect straight lines. Constraints such as orientation and length are used in this straight-line search. The set of detected, nearly-horizontal straight-lines contains (potentially) the desired rooftop edge of the building (Fig. 5e).

Let l_i denote a particular straight line. The building's rooftop edge determination takes into account the following three criteria: first, the length difference between each line l_i and back-projected footprint edge l on I_P, denoted as l_P; second, the horizontal displacement between each line l_i and l_P; and third, the vertical location of each line l_i on I_P.

The determined rooftop edge of the building, denoted as l_t, can then be used to calculate the height of the building using the information of (x_i, y_i) and the location of l in the aerial image.

C. Texture Map Generation

The building's facade image is generated by first back-projecting the color panoramic image region I_R onto a planar surface, denoted by I_Q, as shown in Fig. 5f. The resolution of the color image I_Q is identical to the resolution of the binary image I_P. In general, the desired textured building facade in the resultant image I_Q is not rectangular. Rectangular image textures of building facades are preferred for the texture mapping task. Therefore, image I_Q will be further cropped and rectified according to the detected building rooftop edge l_t and the set of nearly-vertical straight lines as shown in Fig. 5e. Perspective transformation was

performed on image I_Q to obtain a rectangular building texture map as illustrated in Fig. 5g.

VI. EXPERIMENTS

The program was mainly written in MATLAB (Math Works, Natick, MA, USA) and partially in C++. The experiments were performed on a Windows XP (Service Pack 3) operating system running on an Intel(R) Core(TM) i7 CPU 920 2.67 GHz with 3 G of RAM. A Point Grey Ladybug 3 digital video camera was mounted on top of a car and used to capture the spherical panoramic images. The car was moving at an average speed of 35 km/hr, and the camera captured 15 panoramic images per second. In this way, adjacent panoramic images were captured at locations approximately one meter apart. The car was also equipped with a GPS system. The resolution of the recorded panoramic images was 2,048 (width) × 1,024 (height).

The proposed approach was also tested on synthetic image sequences. One reason for this was that there was no sufficiently accurate ground truth available for testing the path recovery approach in the real world. Therefore, we constructed a 12 × 20 unit (a "unit" as used in the program) synthetic city and street environment using commercially available 3D modeling software (Fig. 4a). The buildings were texture-mapped using textures of real buildings. Fig. 4 illustrates a synthetic world experiment on path recovery. In this experiment, 50 panoramic images were generated at locations indicated by the shown dots. The estimated camera path is represented by a set of circles. The average drift of the resulting camera path from the actual path is equal to 0.324 units. This result was not yet based on any bundle adjustment strategy.

Fig. 6 shows two close-up views. A virtual camera was implemented for recording panoramic images in the virtual world. A few examples of captured spherical images are shown in Fig. 7a, which have been transformed into planar rectangular images.

Fig. 6. Two close-up views of a virtual city and streets, modeled by Maya for performing experiments in a synthetic world with accurate ground truth.

(a) (b)

Fig. 7. (a) Panoramic image examples, (b) generated building texture maps.

Fig. 8. Reconstruction examples of 3-dimensional textured buildings.

The aerial image of that area was obtained from Google Maps. Examples of building texture maps, generated from the panoramic images, are shown in Fig. 7b. Fig. 8 illustrates 3D models of buildings obtained semi-automatically by following the discussed approach.

VII. CONCLUSIONS

This paper presented an approach for deriving a realistic looking 3D polyhedral city model. Two types of input images have been used, namely a sequence of spherical panoramic images and a nearly orthogonal aerial image of that area, generated from available aerial image data. Both kinds of images are used cooperatively to recover the camera path and for a subsequent automated extraction of building texture maps from available input sequences. Relative directions and positions of all the panoramic images used were estimated within a fully automatic process with reasonable accuracy, and the camera path was recovered with the help of sparse GPS data.

Moreover, the developed algorithm is able to automatically search (within a recorded panoramic image sequence) for an appropriate initial texture window, and to generate from this a rectangular front-view texture for each building to be reconstructed. At the same time, we can also estimate the height of the building. These capabilities are useful for performing large-scale 3D city modeling.

ACKNOWLEDGMENTS

This project was financially sponsored by the National Science Council (grant no. NSC101-2221-E-197-034 and NSC100-2221-E-197-028). The authors thank Augustine Tsai for support regarding the equipment used for panoramic image acquisitions, and Yi-Ju Wu, Jing-Siang Hsu, and Jui-Yang Tsai for their help in the experiments.

REFERENCES

[1] F. Huang, R. Klette, and K. Scheibe, *Panoramic Imaging: Sensor-Line Cameras and Laser Range-Finders*, Chichester, UK: John Wiley & Sons, 2008.

[2] A. Gruen, "Automation in building reconstruction," in *Photogrammetric Week 1997*, Heidelberg, Germany: Wichmann Verlag, pp. 175-186, 1997.

[3] C. Brenner and N. Haala, "Fast production of virtual reality city models," in *Proceedings of the IAPRS: ISPRS Commission IV Symposium on GIS – Between Visions and Applications*, Stuttgart, Germany, pp. 77-84, 1998.

[4] H. G. Maas, "The suitability of airborne laser scanner data for automatic 3D object reconstruction," in *Automatic Extraction of Man-Made Objects from Aerial and Space Images (III)*, Lisse, Netherlands: A. A. Balkema, pp. 291-296, 2001.

[5] C. Vestri and F. Devernay, "Using robust methods for automatic extraction of buildings," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai: HI, pp. 133-138, 2001.

[6] C. Fruh and A. Zakhor, "3D model generation for cities using aerial photographs and ground level laser scans," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai: HI, pp. 31-38, 2001.

[7] C. Fruh and A. Zakhor, "Constructing 3D city models by merging aerial and ground views," *IEEE Computer Graphics and Applications*, vol. 23, no. 6, pp. 52-61, 2003.

[8] C. Fruh and A. Zakhor, "An automated method for large-scale, ground-based city model acquisition," *International Journal of Computer Vision*, vol. 60, no. 1, pp. 5-24, 2004.

[9] J. Hu, S. You, and U. Neumann, "Automatic pose recovery for high-quality textures generation," in *Proceedings of the 18th International Conference on Pattern Recognition*, Hong Kong, China, pp. 561-565, 2006.

[10] L. Liu, G. Yu, G. Wolberg, and S. Zokai, "Multiview geometry for texture mapping 2D images onto 3D range data," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York: NY, pp. 2293-2300, 2006.

[11] I. Stamos and P. K. Allen, "Geometry and texture recovery of scenes of large scale," *Computer Vision and Image Understanding*, vol. 88, no. 2, pp. 94-118, 2002.

[12] B. Micusik and J. Kosecka, "Piecewise planar city 3D modeling from street view panoramic sequences," in *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition*, Miami: FL, pp. 2906-2912, 2009.

[13] 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," *International Journal of Computer Vision*, vol. 78, no. 2-3, pp. 143-167, 2008.

[14] H. Y. Shum, M. Han, and R. Szeliski, "Interactive construction of 3D models from panoramic mosaics," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Santa Barbara: CA, pp. 427-433, 1998.

[15] L. Wang, S. You, and U. Newmann, "Semiautomatic registration between ground-level panoramas and an orthorectified aerial image for building modeling," in *Proceedings of the IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, pp. 1-8, 2007.

[16] S. B. Kang, R. Szeliski, and M. Uyttendaele, "Seamless stitching using multi-perspective plane sweep," Microsoft Research, Redmond, WA, Technical Report MSR-TR-2004-48, 2004.

[17] D. Lowe, Demo Software: SIFT keypoint detector [Internet], Available: http://www.cs.ubc.ca/~lowe/keypoints/.

[18] C. R. Wren, Perspective transform estimation [Internet], Available: http://alumni.media.mit.edu/~cwren/interpolator/.

Fay Huang

received the M.Sc. Degree with honours (second class, first division) in Pure Mathematics and the Ph.D. degree in Computer Science (2002) from The University of Auckland, New Zealand. In 2003-2004 she was a postdoctoral fellow at the Institute of Information Science, Academic Sinica, Taipei. Since 2006 she is at the Institute of Computer Science and Information Engineering, National Ilan University, Yi-Lan, Taiwan. Her research interests include computer vision and computer graphics, among which she is particularly interested in applications related to virtual reality and computer art.

Reinhard Klette
is a Fellow of the Royal Society of New Zealand and a professor in the Computer Science Department at Auckland University, New Zealand. His professional interests are in computer vision (in theory and applications) and in the design of geometric algorithms. He has (co-)authored more than 350 peer-reviewed publications. He is the Editor-in-Chief of the *Journal on Control Engineering and Technology*, published in Hong Kong.