AFRL-RY-WP-TR-2010-1004

GEO-REFERENCED DYNAMIC PUSHBROOM STEREO MOSAICS FOR 3D AND MOVING TARGET EXTRACTION–A NEW GEOMETRIC APPROACH

Zhigang Zhu, Hao Tang, and Edgardo Molina

City College of New York

DECEMBER 2009
Final Report

Approved for public release; distribution unlimited.

See additional restrictions described on inside pages

STINFO COPY

AIR FORCE RESEARCH LABORATORY
SENSORS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

NOTICE AND SIGNATURE PAGE

AFRL-RY-WP-TR-2010-1004 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

*//Signature//

OLGA MENDOZA-SCHROCK
Program Manager
ATR & Fusion Algorithms Branch
Sensor ATR Technology Division

//Signature//

CHRISTINA G. SCHUTTE, Chief
ATR & Fusion Algorithms Branch
Sensor ATR Technology Division

//Signature//

CHRISTOPHER J. RISTICH
Chief
Sensor ATR Technology Division
Sensors Directorate

*Disseminated copies will show "//Signature//" stamped or typed above the signature blocks.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE ***(DD-MM-YY)*** December 2009	**2. REPORT TYPE** Final	**3. DATES COVERED** ***(From - To)*** 07 March 2005 – 06 September 2009

4. TITLE AND SUBTITLE
GEO-REFERENCED DYNAMIC PUSHBROOM STEREO MOSAICS FOR 3D AND MOVING TARGET EXTRACTION–A NEW GEOMETRIC APPROACH

5a. CONTRACT NUMBER
FA8650-05-1-1853

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62204F

6. AUTHOR(S)
Zhigang Zhu, Hao Tang, and Edgardo Molina

5d. PROJECT NUMBER
6095

5e. TASK NUMBER
04

5f. WORK UNIT NUMBER
60950418

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
City College of New York
Computer Science Department
Convent Avenue and 138th Street
New York, NY 10031

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory
Sensors Directorate
Wright-Patterson Air Force Base, OH 45433-7320
Air Force Materiel Command
United States Air Force

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/RYAT

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-RY-WP-TR-2010-1004

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES
PAO Case Number: 88ABW/10-0154; Clearance Date: 14 Jan 2010. This report contains color.

14. ABSTRACT

We propose a content-based 3D mosaic (CB3M) representation for long video sequences of 3D and dynamic urban scenes captured by a camera on a mobile platform. In the first phase, a set of parallel-perspective (pushbroom) mosaics with varying viewing directions is generated to capture both the 3D and dynamic aspects of the scene under the camera coverage. In the second phase, a segmentation-based stereo matching algorithm is applied to extract parametric representations of the color, structure and motion of the dynamic and/or 3D objects in urban scenes, where a lot of planar surfaces exist. Multiple pairs of stereo mosaics are used for facilitating reliable stereo matching, occlusion handling, accurate 3D reconstruction and robust moving target detection. We use the fact that all the static objects obey the epipolar geometry of pushbroom stereo, whereas an independent moving object either violates the epipolar geometry if the motion is not in the direction of sensor motion or exhibits unusual 3D structures otherwise. The CB3M is a highly compressed visual representation for a dynamic 3D scene, and has object contents of both 3D and motion information. Experimental results are given for both simulated and several different real video sequences of large-scale 3D scenes to show the accuracy and effectiveness of the representation. Applications include airborne or ground video surveillance, 3D urban scene construction, traffic survey and transportation planning. We also discuss the extension of the method to other kind of camera motion.

15. SUBJECT TERMS
Multi-image registration, content-based video coding, image-based modeling, 3D scene representation

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT:**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** (Monitor)
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	SAR	62	Olga Mendoza-Schrock
Unclassified	Unclassified	Unclassified			**19b. TELEPHONE NUMBER** *(Include Area Code)* N/A

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

In addition to the generous support of the Air Force Research Laboratory (AFRL/RYAT) through Grant Award No. FA8650-05-1-1853 (BAA-04-03-SNK Call 9), this work is also partially supported by NSF under Grant No. CNS-0551598, by AFOSR (Award #FA9550-08-1-0199), by ARO (Grant No W911NF-05-1-0011) and by New York Institute for Advanced Studies. Some data were captured in the UMass Computer Vision Laboratory under an NSF grant (No EIA-9726401). The authors are grateful to Ms. Olga Mendoza-Schrock, Dr. Jeff Layne, and Dr. Greg Arnold at AFRL for their guidance and valuable discussions during the period of the project. The authors also want to thank Prof. George Wolberg at CCNY, Prof. Robert Haralick at the CUNY Graduate Center for their involvements in the early stage of the project.

The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. However, the views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government.

1. Introduction

We address the problems of visual representations for large amounts of video stream data, of dynamic three-dimensional (3D) urban scenes, captured by a camera mounted on a low-altitude airborne or a ground mobile platform. Applications include airborne or ground video surveillance for moving target extraction, automated 3D urban scene construction, airborne/ground traffic survey, and image-based modeling and rendering. For these applications, there are two major challenges. First, hours of video streams may be generated every time the mobile platform performs a data collection task. The data amount is in the order of 100 GB per hour for standard 640*480 raw color images. The huge amount of video data not only poses difficulties in data recording and archiving but also is prohibitive for users to retrieve, review or to process. Second, due to the 3D nature of urban scene observed by a moving platform, we will have to naturally and effectively handle obvious motion parallax and object occlusions in order to be able to detect moving objects of interest. Most of the existing algorithms using change detection assuming planar scene or stationary camera will fail in this situation. Compact scene representations and efficient video analysis algorithms are critical for modeling large-scale 3D man-made urban scenes with fine structures, textureless regions, sharp depth changes, and occlusions, as well as moving targets. In applications such as aerial surveillance and transportation planning during an emergency situation, by flying through an area, information such as the location of an abnormal event, the speed, flow and density of the traffic of the entire area, can be immediately calculated and transmitted back to a control center. In addition to the dynamic traffic information, context information about the static objects (buildings, roads and facilities) in the area can also be detected and provided in a highly compressed form. Critical information with large field-of-view coverage can be obtained in a timely and space-efficient manner for immediate decision making.

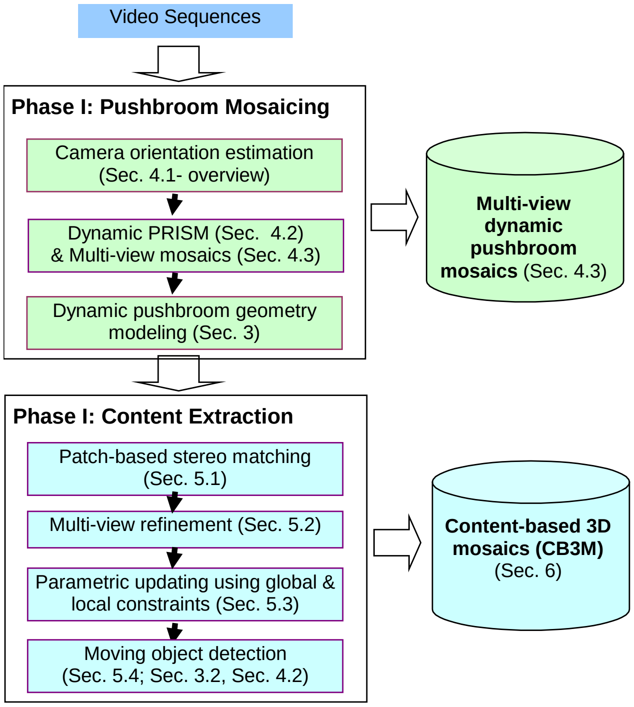
Video Sequences
Phase I: Pushbroom Mosaicing
Camera orientation estimation (Sec. 4.1- overview)
Dynamic PRISM (Sec. 4.2) & Multi-view mosaics (Sec. 4.3)
Dynamic pushbroom geometry modeling (Sec. 3)
Multi-view dynamic pushbroom mosaics (Sec. 4.3)
Phase I: Content Extraction
Patch-based stereo matching (Sec. 5.1)
Multi-view refinement (Sec. 5.2)
Parametric updating using global & local constraints (Sec. 5.3)
Moving object detection (Sec. 5.4; Sec. 3.2, Sec. 4.2)
Content-based 3D mosaics (CB3M) (Sec. 6)

Fig. 1. System diagram

We propose a content-based 3D mosaic representation (CB3M) for long video sequences of 3D and dynamic scenes captured by such a camera mounted on a mobile platform. The motion of the camera has a dominant direction of motion (as on an airplane or ground vehicle), but 6 DOF motion is allowed. We have developed a two-phase procedure for this goal, as shown in Fig.1. In the first phase, a set of parallel-perspective (pushbroom) mosaics with varying viewing directions is generated to capture both the 3D and dynamic aspects of the scene under the camera coverage. Bundle adjustment techniques can be used for camera pose estimation, sometimes integrated with the geo-referenced data from GPS and INS when available. A ray interpolation approach called PRISM (parallel ray interpolation for stereo mosaicing) is used to generate multiple seamless parallel-perspective mosaics under the obvious motion parallax of a translating camera. The set of the multi-view *dynamic* pushbroom mosaics, with a pair of stereo mosaics as the minimum sub-set, is a compact visual representation for a long video sequence of a *3D* scene with independent *moving* targets. In this phase, the epipolar geometry of the multi-perspective pushbroom stereo mosaics is also established to facilitate stereo matching and moving target detection in the next phase.

However, the 2D mosaic representation is still an image-based one without object *content* representations. Therefore, in the second phase, a segmentation-based (“patch-based”) stereo matching approach is proposed to extract parametric representation of the color, structure and motion of the dynamic and/or 3D objects (i.e., the contents) in urban scenes, where a lot of planar surfaces exist. In our approach, we use the fact that all the static objects obey the epipolar geometry, i.e. along the epipolar lines of pushbroom stereo. An independent moving object (moving on a road surface), on the other hand, either violates the epipolar geometry if the motion is not in the direction of sensor motion, or exhibits unusual 3D structure otherwise, e.g., obviously hanging above the road or hiding below the road. Furthermore, multiple pairs of stereo mosaics and local/global spatial constraints are used for facilitating reliable stereo matching, occlusion handling, accurate 3D reconstruction and robust moving target detection.

Based on the above two phases, a *content-based 3D mosaic (CB3M)* representation is created for a long video sequence. This is a highly compressed visual representation for the video sequence of a dynamic 3D scene. For example, a real image sequence of a campus scene has 1000 frames of 640*480 color images. With its CB3M representation, a compression ratio of more than 10,000 is achieved. More importantly, the CB3M representation has high-level object *contents*. A scene is represented in parametric forms of planar regions with their 3D, their boundaries, their motion, and their relations. The CB3M representation can be utilized for object recognition and indexing.

There are three technical challenges in generating a content-based 3D mosaic representation from a long image sequence. They are (1) robust and accurate camera orientation estimation for many video frames; (2) seamless video mosaic generation with obvious motion parallax; and (3) accurate 3D reconstruction for large-scale urban scenes. In our previous study (Zhu, et al, 2004), we have proposed an algorithm, called parallel ray interpolation for stereo mosaicing (PRISM) that generates seamless mosaics under motion parallax, for static scenes. In another piece of work (Zhu, et al, 2003), we proved by theoretical analysis that with parallel-perspective stereo mosaic, depth error is constant in theory and is linearly proportional to depth in practice. We have also implemented practical methods in camera orientation estimation with external orientation measurements (Zhu, et al, 2005).

Based on the previous work, we have made the following three significant new contributions.

First, we extend the previous work on stereo mosaics from static scenes to dynamic scenes, thus allowing the handling of independent moving objects. This is significant in low-altitude aerial video surveillance of urban scenes since traditional methods using change detection fail to work here due to motion parallax. We also show that the PRISM algorithm also works for dynamic scenes, which means we can re-use the code we have developed for stereo mosaics of static scenes. These results are mainly presented in Sections 3 and 4.

Second, an effective and efficient patch-based stereo matching method has been proposed to extract both 3D and motion information from stereo mosaics of urban scenes, which feature sharp depth boundaries and many textureless regions. This is a unified approach for both 3D reconstruction and moving target extraction. Furthermore, this method can produce higher-level scene representations rather than just depth maps, which leads to our highly compressed content-based video representation. Note that in our previous work, we had only used correlation-based stereo matching methods successfully for highly textured scenes such as forestry scenes, which does not work well with urban scenes with sharp depth boundaries and many textureless regions. In addition, the new approach can also be used with other stereo geometry. These are mainly discussed in Section 5 and 6.

Finally, we perform thorough experimental analysis of the robustness and accuracy of 3D reconstruction using parallel-perspective stereo mosaics. We show the high accuracy of 3D reconstruction and moving target detection by using a simulated video sequence while both ground truth data of 3D urban model and accurate camera orientation information are available, which motivates us and other researchers for developing robust and efficient algorithms to estimate camera orientation with many image frames. On the other hand, using a simplified camera orientation estimation method for several real-world video sequences, we have found that we can generate very compelling stereo perception and reliable 3D depth information. This indicates that for some applications where accurate 3D measurements are not critical, such as image-based rendering, and even automatic target detection and transportation analysis, we can ease the challenging problem of many-frame camera orientation estimation. The experimental analysis is mainly discussed in Section 7.

The rest of the report is organized as follows. Section 2 discusses some related work. In Section 3, the mathematical framework of the dynamic pushbroom stereo is given, and then its properties for moving target extraction are discussed. In Section 4, technical issues of dynamic stereo mosaics in real-world applications are discussed, and multi-view pushbroom mosaics are proposed for image-based rendering and for extracting 3D structure and moving targets. In Section 5, our multi-view pushbroom stereo matching approach for 3D reconstruction and moving target extraction is provided. Then in Section 6, the content-based 3D mosaic representation is described. Experimental results of CB3M representation construction will be given in Section 7 with both simulated and several very different video sequences of both outdoor and indoor 3D scenes. Section 8 discusses image-based rendering based on both stereo mosaics and content-based 3D mosaics. In Section 9, we generalize the model to the case of circular camera motion and provide some preliminary results. Section 10 gives concluding remarks and discusses some future research directions.

2. Related work

Reconstructing and representing large-scale 3D scenes from multiple images has attracted a lot of attention for quite some time. For example, the work at CMU (Herman & Kanade, 1984; Herman & Kanade, 1986) represents one of the first efforts in incrementally constructing 3D scenes from multiple complex images. Interestingly, they used “3D MOSAIC” as the name of their system. However, it is the advancement of both hardware and software in the last ten to fifteen years that makes it possible to efficiently process huge amounts of video data and to generate panoramic mosaics using a general purpose PC. Since then, mosaics have become common for combining and representing a set of images gathered by one moving camera or multiple cameras. In the past, video mosaic approaches (Irani, et al, 1996; Hsu & Anandan, 1996; Odone, et al, 2000; Leung & Chen, 2000) have been proposed for video representation and compression, but most of the work is for generating 2D mosaics instead of 3D panoramas, and using panning (rotating) cameras for arbitrary scenes or moving cameras for planar scenes, instead of traveling (translating) cameras typically used in airborne or ground mobile urban surveillance and 3D scene modeling. In the latter applications, obvious motion parallax is the main characterization of the video sequences due to the self-motion of the sensors and obvious depth changes of the scenes.

To generate truly “3D mosaics” from video sequences of a traveling camera, we are particularly interested in the parallel-perspective *pushbroom stereo* geometry (Chai & Shum, 2000; Zhu, et al, 2004). The term “pushbroom” is borrowed from satellite pushbroom imaging (Gupta & Hartley, 1997) where a linear pushbroom camera is used. The basic idea of the pushbroom stereo mosaics is as follows. If we assume the motion of a camera is a 1D translation and the optical axis is perpendicular to the motion, then we can generate two spatio-temporal images (mosaics) by extracting two scanlines of pixels of each frame (perpendicular to the motion of the camera), one in the leading edge and the other in the trailing edge. Each mosaic image thus generated is similar to a *parallel-perspective* image captured by a linear pushbroom camera, which has parallel projection in the direction of the camera’s motion and perspective projection in the direction perpendicular to that motion. Pushbroom stereo mosaics have uniform depth resolution, which is better than with perspective stereo, and the multi-perspective stereo with circular projection (Peleg, et al 2001; Shum & Szeliski, 1999). Pushbroom stereo mosaics can be used in applications where the motion of the camera has a dominant translational direction. Examples include satellite pushbroom imaging (Gupta & Hartley, 1997), airborne video surveillance (Zhu, et al, 2004), image-based rendering with 3D reconstruction or 3D estimation (Chai & Shum, 2000, Rav-Acha, et al, 2008), 3D representations of ground route scenes (Zheng & Tsuji, 1992; Zhu & Hanson, 2004, Zheng & Shi, 2008), under-vehicle inspection (Dickson, et al, 2002; Koschan, et al, 2004), 3D measurements of industrial parts by an X-ray scanning system (Noble, et al, 1994), and 3D gamma-ray cargo inspection (Zhu & Hu, 2007). Some work has been done in 3D reconstruction of panoramic mosaics (Li, et al, 2004; Sun & Peleg, 2004) with an off-center rotation camera, but the methods are limited to a fixed view-point camera instead of a moving camera, and usually the results are still low-level 3D depth maps of *static* scenes, instead of high-level 3D structural representations for both static and *dynamic* target extraction and indexing. On the other hand, layered representations (e.g., Xiao & Shah, 2004; Zhou & Tao, 2003; Ke & Kanade, 2001) have been studied for motion sequence representations; however, the methods are usually computationally expensive, and the outputs are typically motion segmentation represented by affine planes instead of true 3D information. Efficient, high-level, content-based, and very low bit-rate representations of videos of 3D scenes and moving targets are still in great demand.

Another class of related work is 3D reconstruction from stereo pairs. Stereo vision is one of the most important topics in computer vision, and recently a thorough comparison study (Scharstein &. Szeliski, 2002) has been performed. Simple window-based correlation approaches do not work well for man-made scenes. In the past, an adaptive window approach (Kanade & Okutomi, 1991) and a nine-window approach (Fusiello, et al, 1997) have been used to deal with some of these issues. Recently, color segmentation has been used for refining an initial depth map to get sharp depth boundaries and to obtain depth values for textureless areas (e.g., Tao, et al, 2001), and for accurate layer extraction (e.g., Ke & Kanade, 2001). Global optimization based stereo matching methods, such as belief propagation (Sun, et al, 2003) and graph cuts (Boykov, et al, 2001; Kolmogorov & Zabih, 2001), can obtain accurate depth information, but these methods are computationally expensive. Cornelis, et al (2008) present a complete system for turning forward-looking stereo video from a moving car into a model from which a virtual drive-through of a city street can be rendered. The paper by Pollefeys, et al (2008) describes a system for automatic, geo-registered, real-time 3D reconstruction from video of urban scenes using a multi-view stereo approach. Most stereo reconstruction papers are based on perspective stereo geometry, except a few papers (Li, et al, 2004; Sun & Peleg, 2004; Zhu & Hanson, 2004) dealing with multi-perspective stereo images.

3. Dynamic Pushbroom Stereo Mosaic Geometry

Stereo mosaics of static scenes have been well-studied in the past. As a preparation, we give a brief description of the concept. Assume the motion of a camera is an ideal 1D translation, the optical axis is perpendicular to the motion, and the frames are dense enough. Then, we can generate two spatio-temporal images by extracting two columns of pixels (perpendicular to the motion) at the leading and trailing edges of each frame in motion. The geometry in this ideal case (i.e. 1D translation with constant speed) is the same as the linear pushbroom camera model (Gupta & Hartley, 1997). Therefore we also call this image representation *pushbroom stereo mosaic representation*. A generalized model under 3D translation (Zhu, et al 2004) has extended the parallel-perspective stereo geometry to image sequences with 3D translation and further with 6 DOF motion (rotation + translation). Here, we will use the parallel-perspective stereo geometry under 1D translation to introduce the new concept of the *dynamic* stereo mosaics.

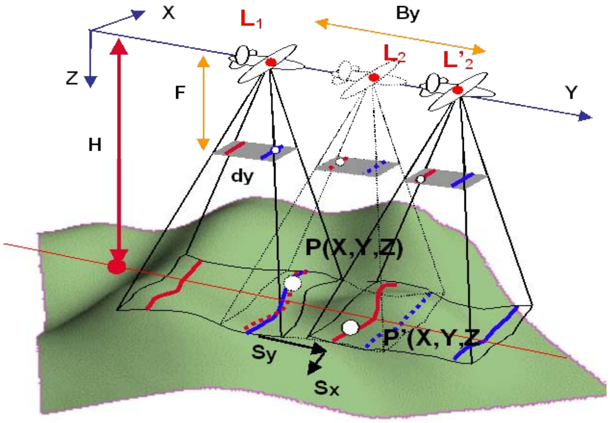

Fig. 2. Dynamic pushbroom stereo mosaics

3.1. Dynamic pushbroom stereo model

For completeness, we start with the formulation of the pushbroom stereo mosaics in a static scene. Without loss of generality, we assume that two slit windows of two scanline locations have d_{yl} and d_{yr} offsets to the center of the image, respectively, and the distance between the two windows is the fixed "disparity" $d_y = d_{yl} - d_{yr} > 0$ (in Fig. 2, $d_{yl} = d_y/2$, $d_{yr} = -d_y/2$). The "left eye" view (x_l,y_l) is generated from the front slit window d_{yl}, while the "right eye" view (x_r, y_r) is generated from the rear slit window d_{yr}. A static point P (X,Y,Z) can be viewed twice from the two slit windows, at the camera location L_1 and L_2, respectively. Then the *parallel-perspective" pushbroom" model* of the stereo mosaics thus generated can be represented by

$$\begin{cases} x_l = x_r = F\dfrac{X}{Z} \\ y_l = F\dfrac{Y}{H} - (\dfrac{Z}{H} - 1)d_{yl} \\ y_r = F\dfrac{Y}{H} - (\dfrac{Z}{H} - 1)d_{yr} \end{cases} \qquad (1)$$

where F is the focal length of the camera, H is the height of a *fixation plane* on which we want to align our stereo mosaics. Eq. (1) gives the relation between a pair of 2D points, (x_l,y_l) and (x_r,y_r), one from each mosaic, and their corresponding 3D point P (X,Y,Z). It serves a function similar to the classical pin-hole perspective camera model. From Eq. (1) the depth of the point P can be computed as

$$Z = H\frac{b_y}{d_y} = H(1+\frac{\Delta y}{d_y}) \tag{2}$$

where $b_y = d_y + \Delta y = F\frac{B_y}{H}$ is the "scaled" version (in pixel) of the “baseline” B_y, i.e., the distance between two camera locations, and

$$\Delta y = y_r - y_l \tag{3}$$

is the "mosaic displacement" in the stereo mosaics. We use “displacement” instead of “disparity” since it is related to the baseline in a two view-perspective stereo system. Displacement Δy is a function of the depth variation of the scene around the fixation plane H. Since a fixed angle between the two viewing rays is selected for generating the stereo mosaics, the "disparities" (d_y) of all points are fixed; instead geometry of optimal/adaptive baselines (b_y) for all the points is created. In other words, for any point in the left mosaic, searching for the match point in the right mosaic means (virtually) finding an original image frame in which the match pair has a pre-defined disparity (by the distance of the two slit windows) and hence has an adaptive baseline depending on the depth of the point. Therefore, a stereo geometry with uniform depth resolution is achieved. More in-depth analysis on depth accuracy of stereo mosaics from real image sequences can be found in our previous paper (Zhu, et al, 2003). In this paper, we focus more on the dynamic aspect of stereo mosaics, and algorithms for simultaneous 3D reconstruction and moving target detection in urban scenes.

Interestingly, dynamic pushbroom stereo mosaics are generated in the same way as with the static pushbroom stereo mosaics described above. Fig. 2 also illustrates the geometry. A 3D point P (X,Y,Z) on a target is first seen through the leading edge (the front slit window) of an image frame when the camera is at location L_1. As we have discussed, if the point P is static, we can expect to see it through the trailing edge (rear slit window) of an image frame when the camera is at location L_2. However, if the point P moves during that time, the camera needs to be at a different location L'_2 to see this moving point through its trailing edge. To simplify the equations, we assume that the motion of the moving point between two observations (L_1 and L'_2) is a 2D motion (S_x, S_y), which implies that the depth of the point does not change over that period of time. Therefore, the depth of the moving point can be calculated as

$$Z = F\frac{B_y - S_y}{d_y} \tag{4}$$

where B_y now is denoted as the distance of the two camera locations (L_1 and L'_2 in the y direction). Mapping this relation into the stereo mosaic notation above (Eq. (2)), we have

$$Z = H(1+\frac{\Delta y - s_y}{d_y}) \tag{5}$$

and

$$(S_x, S_y) = (Z\frac{s_x}{F}, H\frac{s_y}{F}) = (Z\frac{\Delta x}{F}, H\frac{s_y}{F}) \tag{6}$$

where □(Δx, □Δy) is the visual motion of the moving 3D point P, which can be measured in the stereo mosaics. The vector (s_x, s_y) is the target motion represented in stereo mosaics. Obviously, we have $s_x = \Delta x$. The above analysis only shows the geometry of a moving camera with 1D translational motion. A pair of generalized stereo mosaics can be generated when the camera undertakes constrained 6 DOF motion, similar to the case of static scenes (Zhu, et al, 2004).

3.2. Moving object extraction against parallax

We have the following interesting observations about the *dynamic* pushbroom stereo geometry for 3D and moving target extraction when obvious motion parallax exists in videos of 3D urban scenes.

(1) *Stereo fixation.* For a static point (i.e. $S_x = S_y = 0$), the visual displacements of the point with a depth H are (0,0), indicating that the stereo mosaics thus generated fixate on the plane of depth H. If the fixation plane is the ground plane, this fixation facilitates stereo matching and moving target detection since the major background (i.e., the ground plane) has been aligned.

(2) *Motion accumulation*. For a moving point ($S_x \neq 0$ and/or $S_y \neq 0$), the motion between two observations accumulates over a period of time due to the large distance between the leading and trailing edges in creating the stereo mosaics. This will increase the discrimination capability for slowly moving objects viewed from a relatively fast moving aerial camera. Typically, a moving object as recorded in a pair of stereo mosaics is originally viewed from two views that are many frames apart (Fig. 2).

(3) *Epipolar constraints.* In the ideal case of 1D translation of the camera (with which we present our dynamic pushbroom stereo geometry in this paper), the correspondences of static points are along horizontal epipolar lines in a pair of pushbroom mosaics, i.e., $\Delta x = 0$. Therefore, for a moving target P, the visual motion with nonzero Δx (i.e., the visual motion in the x direction) will identify itself from the static background in the general case, which implies that the motion of the target in the x direction is not zero (i.e., $S_x \neq 0$). In other words, the correspondence pair of such a point will violate the epipolar line constraint for static points (i.e. $\Delta x = 0$). Note that this represents the general cases of independent moving targets.

(4) *3D constraints.* Even if the motion of the target happens to be in the direction of the camera's motion (i.e., the y direction), we can still discriminate the moving target by examining 3D anomalies. Typically, a moving target (a vehicle or a human) moves on a flat ground surface (i.e., road) over the time period during which it is observed through the leading and trailing edges of video images with a limited field of view. We can usually assume that the moving target shares the same depth as its surroundings, given that the distance of the camera from the ground is much larger than the height of the target. A moving target in the direction of camera movement, when treated as a static target, will show 3D anomaly - either hanging up above the road (when it moves to the opposite direction, i.e., $S_y < 0$), or hiding below the road (when it moves in the same direction, i.e., $S_y > 0$). Note this is only the special case of independent moving targets.

After a moving target has been identified, the motion parameters of the moving target can be estimated. We first estimate the depth of its surroundings and apply this depth Z to the target, then calculate the object motion s_y using Eq. (5), and (S_x, S_y) using Eq. (6), knowing the visual motion (Δx, Δy) measured in the stereo mosaics.

4. Real-World Issues and Multi-View Mosaics

In real applications, there are three sets of challenging problems. These include camera motion estimation in practical cases, mosaic generation with more general camera motion, and occlusion and stereo matching issues in a pair of stereo mosaics. For some issues, we will give very brief discussions and point to related work. More details will be given for dynamic stereo mosaic generation, and multi-view pushbroom mosaics for dealing with occlusions, stereo matching and moving target detection.

4.1. Camera orientation estimation

The first problem is that the camera usually cannot be controlled with ideal 1D translation and camera poses are unknown; therefore, camera orientation estimation (i.e., dynamic calibration) is needed. In our previous study on an aerial video application, we used external orientation instruments, i.e., GPS, INS and a laser profiler, to ease the problem of camera orientation estimation (Zhu, et al, 2004; Zhu et al 2005). More general approaches using bundle adjustment techniques (Triggs, et al, 2000) are under investigation for estimating camera poses of long image sequences, which is one of the challenging issues of our stereo mosaic approach, and of video sequence analysis in general. In this paper, we focus on other technical issues of the problem, and use an ideal 1D camera translational model to show the principle of the dynamic pushbroom stereo mosaics, without loss of generality. In our experimental analysis, we either assume that the extrinsic and intrinsic camera parameters are known at each camera location, as in theoretical analysis, or use a simplified version of camera orientation estimation, in which only four camera parameters are used. The four parameters are translation components in the X and Y directions, a heading angle, and a scaling factor. An underlying assumption in the practical treatments is that, (1) if the translational component in the Z direction is much smaller than the distance itself, we use a constant scaling factor in the interframe motion estimation and image rectification for each frame to compensate for the Z translation; and (2) the rolling and tilting angles are small so they are combined into the translations in the X and Y directions. The mosaics from real video sequences are generated from such camera orientation estimation model. We have found that 3D perception is compelling and 3D reconstruction results are reliable with such treatments, and the results could still be useful for image-based rendering and automated target detection.

4.2. Stereo mosaicing for dynamic scenes

The second problem is to generate dense parallel mosaics with a sparse, uneven, video sequence, under a more general motion, and for a complicated 3D scene. For the case of static scenes, we have proposed a parallel ray interpolation for stereo mosaics (PRISM) approach (Zhu, et al 2004) for generating a generalized stereo mosaic representation for static scenes, under constrained 6 DOF motion. At the first look, the approach might not be applicable to dynamic scenes. But a careful study shows that the PRISM approach designed for static scenes also works for dynamic scenes. Fig. 3 illustrates the basic idea of the PRISM algorithm in generating one forward-looking *dynamic* pushbroom mosaic (left mosaic with slit window location d_{yl}). In the figure, (T_{x1}, T_{y1}, T_{z1}) and (T_{x2}, T_{y2}, T_{z2}) denote two consecutive camera locations, at time t_1 and t_2, respectively. From each of the two frames, only one scan line (the fixed line) can be directly used for the mosaic since it is generated from the correct viewing direction. For any other point P between these two fixed lines, its parallel-perspective projection needs to be interpolated from its matching pair in the two frames, (x_1, y_1) and (x_2, y_2), respectively. If the point P is a static point, the triangulation gives its correct 3D location $P(X,Y,Z)$, and its backprojection gives the necessary parallel view as seen from the "interpolated" camera location (T_{xi}, T_{yi}, T_{zi}), where

$$T_{yi} = T_{y1} + \frac{y_1 - d_{yl}}{y_1 - y_2}(T_{y2} - T_{y1}), T_{xi} = T_{xl}, T_{zi} = T_{zl} \tag{7}$$

assuming $T_{x1} = T_{x2}$ and $T_{z1} = T_{z2}$ under the ideal 1D camera motion case. However, for a moving point (from 3D positions P_{t1} to P_{t2}), the triangulation does not give us its right 3D coordinates, but the back-projection will create an image of the moving point P_{ti} that should be seen at the "interpolated" time t_i , i.e. at camera location (T_{xi} , T_{yi} , T_{zi}), which is a linear interpolation between time t_1 and t_2. This naturally gives a linearly pushbroom scanning of the moving point. Under the linear motion assumption, the mosaic coordinates of the pair of point are

$$y_i = \frac{F}{H} T_{yi} + d_{y1}, x_i = x_1 \tag{8}$$

This is an important finding since the mosaicing algorithms developed for static scenes can be directly applied to dynamic scenes.

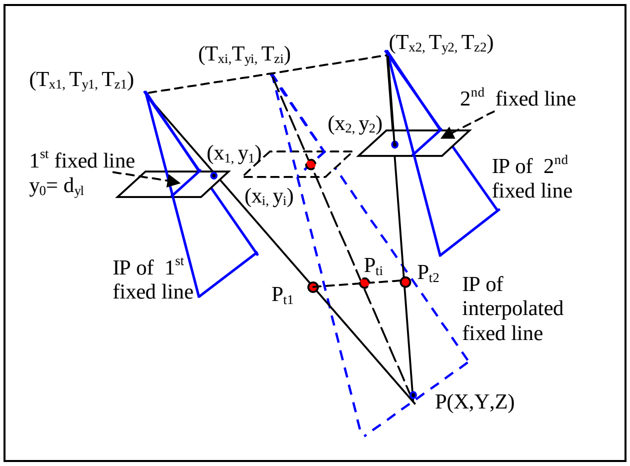

Fig. 3. Ray interpolation for a dynamic scene

In principle, the PRISM approach needs to match all the points between the two overlapping slices of the successive frames to generate a complete parallel-perspective mosaic. In an effort to reduce the computational complexity, a fast PRISM algorithm has been designed (Zhu, et al 2004), based on the proposed PRISM method. It only requires matches between a set of control point pairs in two successive images, and the rest of the points are generated by warping a set of triangulated regions defined by the control points in each of the two images. The proposed fast PRISM algorithm can be easily extended to use more feature points (thus smaller triangles) in the overlapping slices so that each triangle really covers a planar patch or a patch that is visually indistinguishable from a planar patch, or to perform pixel-wise dense matches to achieve true parallel-perspective (pushbroom) geometry.

4.3. Multi-view pushbroom mosaics for dynamic scenes

Finally, 3D reconstruction and motion detection from two widely separated stereo mosaics raise challenging issues. A pair of stereo mosaics (generated from the leading and trailing edges) is a very efficient representation for both 3D structures and target movements. However, there are two remaining issues. First, stereo matching will be difficult due to the largely separated parallel views of the stereo pair, resulting in large perspective distortions and varying occlusions. Second, for some unusual target movements, e.g. moving too fast, changing speed or direction, we may either have two rather different images in the two mosaics (if changing speed), or we see the object only once (if changing direction), or we never see the object (if it maintains the same speed as the camera and thus never shows up in the second edge window).

Therefore, we propose to generate multi-view mosaics (more than 2), each of them with a set of parallel rays whose viewing direction d_{yk} is between the leading and the trailing edges, d_{y0} and d_{yK}, respectively (Fig. 4, k = 0, 1, …, K). The multiple mosaic representation is still efficient. Moreover, there are three benefits of using them. First, multiple pushbroom mosaics can be used for image-based rendering with stereo viewing in which the translation across the area is simply a shift of a pair of mosaics, and the change of viewing directions is simply a switch between two consecutive pairs of mosaics. We will give a brief discussion on image-based rendering using multiple stereo mosaics in Section 8. Second, it eases the stereo correspondence problem in the same way as the multi-baseline stereo (Okutomi & Kanade, 1993), particularly for more accurate 3D estimation and occlusion handling. In the stack of pushbroom mosaics, different sides of a 3D object will be represented in mosaics with various viewing angles. Each of these mosaics with parallel projections views the scene from a unique parallel viewing direction, thus captures surfaces of 3D objects visible from that direction (refer to Fig. 10 a-c for three views of pushbroom mosaics with different sides of buildings visible in different mosaics). In the next section, we will discuss in details a new method to extract both the 3D structures and moving targets from multiple dynamic pushbroom mosaics. We will also discuss the possibility to extract and represent occluding regions in Section 6.

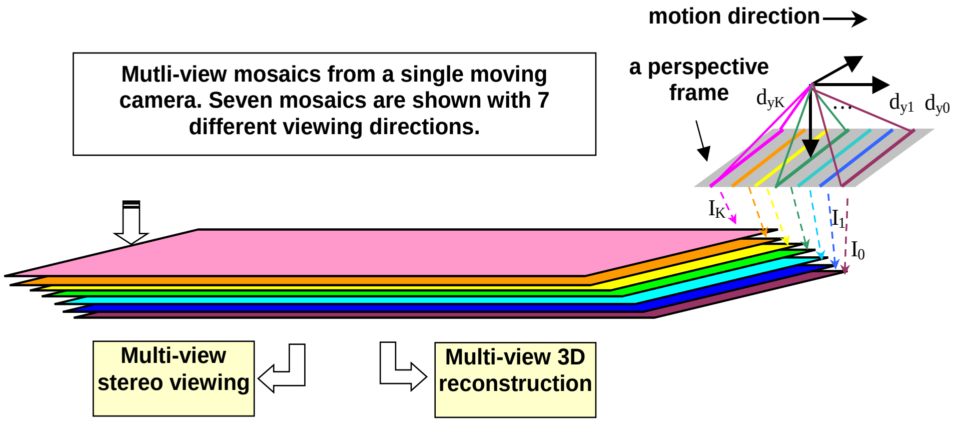
motion direction
Mutli-view mosaics from a single moving camera. Seven mosaics are shown with 7 different viewing directions.
a perspective frame
d_{yK}
…
d_{y1}
d_{y0}
I_K
I_1
I_0
Multi-view stereo viewing
Multi-view 3D reconstruction

Fig. 4. Multi-view pushbroom mosaics

Third, multiple mosaics can also facilitate 3D estimation of moving targets, and increase the possibility to detect moving targets with unusual movements and also to distinguish the movements of the specified targets (e.g., ground vehicles) from those of trees or flags in wind. Here we want to briefly discuss how multi-view mosaics can be used to estimate 3D structure of a moving target on the ground. In order to estimate the height of a moving target from the ground, we will need to see both the bottom and the top of an object. A pair of pushbroom mosaics with one forward-looking view and the other backward-looking view exhibits obvious different occlusions; in particular, the bottom of a target (e.g., a vehicle in Fig. 5a) can only be seen in one of the two views. However, any two of the multi-view pushbroom mosaics, if both with forward-looking (or backward-looking) parallel rays, will have almost the same occlusion relation to satisfy the condition for height estimation.

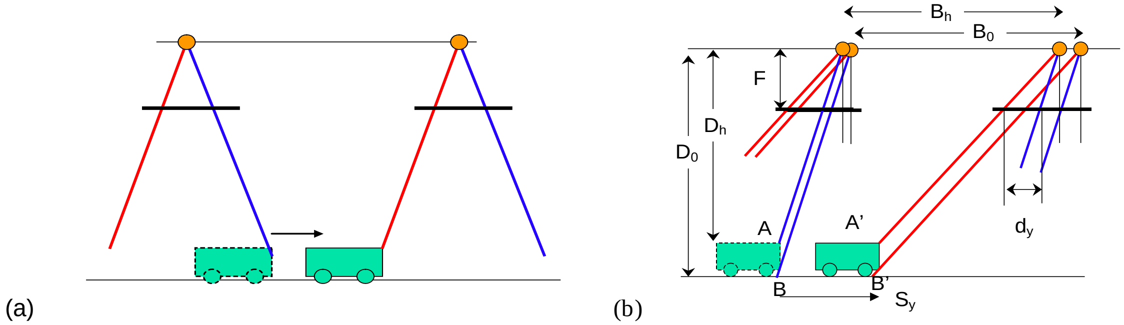

Fig. 5. Height from dynamic pushbroom stereo: (a) an infeasible pair; (b) a feasible pair

Fig. 5b illustrates the case of a pair of backward-looking pushbroom stereo mosaics. Point *A* and *B* are two points on a target (vehicle), one on the top and the other on the bottom. Both of them are first seen in the mosaic with parallel rays of a smaller oblique angle, and then seen in the mosaic with parallel rays of a larger oblique angle. The distance between the two different rays within an image frame is still defined as d_y. The visual motion in the y direction is Δy_h and $\Delta y_{0,}$ respectively, and can be measured in the stereo pair. Between the two parallel views, let us assume the motion of the target is S_y in 3D space and s_y in the mosaiced images. Then the depths of the points on the top and on the bottom are

$$Z_h = F\frac{B_h - S_y}{d_y} = H(\frac{d_y + \Delta y_h - s_y}{d_y}) \tag{9}$$

and

$$Z_0 = F\frac{B_0 - S_y}{d_y} = H(\frac{d_y + \Delta y_0 - s_y}{d_y}) \tag{10}$$

respectively. Depth Z_0 of the bottom point could be obtained from the surroundings (ground) of the target. Then, the object motion s_y (and therefore S_y) can be calculated using Eq. (10). Finally, the depth of the point on the top, Z_h, can be estimated using Eq. (9), given the known visual motion of that point, Δy_h, and its independent motion component s_y obtained from the bottom point *B*.

5. 3D and Motion Content Extraction

Using the advantageous properties of multi-view mosaics, we propose a unified approach to perform both stereo matching and motion detection. In a set of pushbroom mosaics, I_0, I_1, …, I_K, generated from a video sequence, at slit window locations d_{y0}, d_{y1}, …, d_{yK} (see Fig. 4), the leftmost mosaic I_0 at the location d_{y0} is used as the reference view, therefore color segmentation is performed on this mosaic, and the so called *natural matching primitives* (explained below) are extracted. Multiple natural matching primitives are defined with each homogeneous color image patch, which approximately corresponds to a planar patch in 3D. The representations are effective for both static and moving targets in man-made urban scenes with objects of largely textureless regions and sharp depth boundaries. Then matches of those natural matching primitives are searched in the rest of the mosaics, one by one. After matching each stereo pair, a plane is fitted for each patch, and its planar parameters are estimated. Then, multi-view matches are performed, and therefore multiple sets of parametric estimates for this planar patch are obtained. The best set is selected as the final result by comparing match evaluation scores. Local and global spatial constraints are also explored to improve the robustness of the 3D estimation. The moving targets are detected after the "3D alignments" of the scene.

The multi-view dynamic stereo mosaic approach has the following four stages: (1) natural patch-based stereo matching; (2) plane estimation from multiple views; (3) plane merging and updating using local and global scene constraints; and (4) moving object extraction using the dynamic pushbroom stereo geometry. We will describe the approach in detail in the following subsections.

5.1. Patch-based stereo matching

Stereo matching is applied first on a pair of stereo mosaics. Let the leftmost (i.e., reference) mosaic and the second mosaic be denoted as I_0 and I_1, respectively. First, the reference mosaic I_0 is segmented into homogeneous color image patches. In our current implementation, the mean-shift-based approach (Comanicu & Meer, 2002) is used; but other segmentation methods can also be used for this purpose. In practice, over-segmentation (into small patches) is undertaken for ensuring homogeneity of each patch to enable accurate 3D recovery; however, a segmentation with larger patches will result in higher compression ratio of the video sequence.

The segmented image consists of image regions (patches), $\{\mathbf{R}_i$, i =1, …, N$\}$, each with a homogeneous color $\mathbf{c}_i$ and is assumed to be a planar region in 3D space. All the neighboring patches, $\{\mathbf{R}_{ij}$, j =1, …, J$\}$, are also recorded for each patch $\mathbf{R}_i$, The boundary of each patch, **b**i, is extracted as a closed curve. Then we use a line fitting approach to extract feature points for stereo matching. The boundary of each patch is first fitted with connected straight-line segments using an iterative curve splitting method. The connecting points between line segments are defined as *interest points*, $\mathbf{p}_{il}$, = 1, …, L, around which the natural matching primitives are defined.

For each interest point, the best match between the reference and target mosaics is searched within a preset search range. Instead of using the conventional window-based match, we define the so-called *natural matching primitives* (Fig. 6) to conduct a sub-pixel stereo match. Note that the natural matching primitives around the detected interest points, instead of line segments or the patches, are the features to be matched. We define a region mask W_l of size w×w centered at each interest point $\mathbf{p}_{il}$ = (x,y) $\in\mathbf{R}_i$, such that

$$W_l(u,v)=\begin{cases}1, if\ (x+u,y+v)\in \mathbf{R}_i\\ 0, otherwise\end{cases} \tag{11}$$

The size w of the mask is adaptively changed depending on the actual size of the region $\mathbf{R_i}$. In order that a few more pixels (1-2) around the region boundary (but not belonging to the region) are also included so that we have sufficient salient image features to match, a dilation operation is applied to the mask W_l to generate a region mask covering pixels across the depth boundary. Fig. 6 shows four such windows for the four interest points for the top region of the box. Note the yellow-shaded portions within each rectangular window, i.e., the natural matching primitives, indicating that the pixels for stereo matching cover the depth boundaries. They are called “natural matching primitives”, because these primitives define the natural structures of the salient visual features, in terms of sizes, shapes and locations. Each natural matching primitive in the reference image is defined by its location (x,y) on the patch’s boundary $\mathbf{b}_i$, and the pixels belonging to the patch, which is represented by the size of a rectangular window and the mask (together they form a “natural” window as a yellow region in Fig 6). To this point, the attributes of each region (patch) $\mathbf{R}_i$ can be summarized as:

$$\mathbf{R}_i=(\mathbf{c}_i,\mathbf{b}_i,\{\mathbf{R}_{i\,j}, j=1,...,J\},\{\mathbf{p}_{il},W_l,l=1,...,L\}), i=1,...,N \tag{12}$$

which includes its color, boundary, J neighboring regions, L interest points and the corresponding masks.

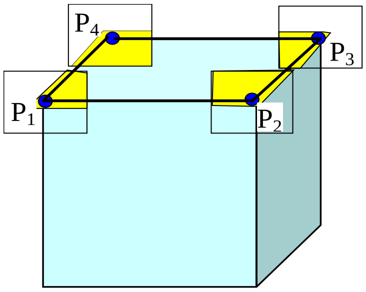

Fig. 6. Natural matching primitives

The weighted cross-correlation, based on the natural window centered at the interest point *(x, y)* in the reference mosaic, is defined as

$$C(\Delta x,\Delta y)=\frac{\sum_{u,v} W_l(u,v)I_0(x+u,y+v)I_1(x+u+\Delta x,y+v+\Delta y)}{\sum_{u,v} W_l(u,v)} \tag{13}$$

Note that we still carry out correlation between two color images but only for those interest points on each region boundary, and for each interest point, the calculation is only carried out on those pixels within the region and on the boundaries. A sub-pixel search is performed in order to improve the accuracy of 3D reconstruction; and a match is marked as *reliable* if it passes the crosscheck (e.g., as in Scharstein & Szeliski, 2002), i.e. the matches from the reference to the target and from the target to the reference are consistent. For the simplicity of representation, we still use Eq. (12) to represent the region $\mathbf{R}_i$, with a note that the number (L) of reliable interest points used in the following steps may be smaller than the total number of interest points.

The matching process consists of the following two steps.

Step 1: *local match*. For each interest point, in order to find a reliable corresponding point, the natural matching strategy is carried out with a multi-scale approach, in that the search ranges and search steps are changed adaptively (from large to small). First, the natural matching strategy is applied to each interest point $\mathbf{p}_{il}$ (l=1...,L) of a region (patch) $\mathbf{R}_i$ (i=1,…, N) in the reference I_0, within preset (large) search range (S_h, S_v) in both the horizontal (y) and vertical (x) directions, and a preset (large) search step *s*. Note that the pushbroom stereo geometry produces image displacement in the y direction, but to account for camera calibration and orientation estimation error, a search within a much smaller range in the x direction is also performed. If a reliable match is obtained, then a new set of parameters (S_h, S_v and *s*) are calculated based on the first run; in other words, the search range is narrowed to the neighborhood of corresponding points with a finer step, and hence S_h, S_v and *s* are reduced. Then, the natural matching procedure is applied again, with the updated parameters. The same procedure is carried out recursively until convergence, i.e., *s* become a fraction (therefore match results are sub-pixel accurate). Usually the match procedure converges in three iteration steps.

Step 2: *Surface fitting*. Assuming that each homogeneous color region $\mathbf{R}_i$ is planar in 3D, then a 3D plane can be generated as

$$a_iX+b_iY+c_iZ=d_i \qquad (14)$$

which is represented in the camera coordinate system as shown in Fig. 2, is fitted to each region after obtaining the 3D coordinates of the interest points of the region using the pushbroom stereo geometry (Eqs. 1 and 2).
We use a standard RANSAC method (e.g., Medioni & Kang, 2004) to fit planes. In our implementation, a plane is fitted by randomly selecting three reliable interest points, and then using the plane parameters, all reliable interest points are warped from the reference view onto the target view. For each reliable interest point, the distance between the warped interest point and its corresponding target point (from local match) is calculated, and if the distance is less than 1 pixel, the point is claimed to be the one that supports the fitted plane. The total number of supports is denoted as C, and the RANSAC process stops if C/L is larger than 65%, where L is the total number of reliable interest points. The number of the random selections of three points is set to N_{max} = 50. In other words, the RANSAC process will stop either at 50 iterations or when the number of the supporting reliable points exceeds 65% of total reliable points. Then the best set of the plane parameters is selected as the initial 3D estimation of the planar patch. In the latter case, the region is marked as a *reliable* patch (in 3D estimation), therefore a unreliable patch at this point is the one whose number of reliable interest points is smaller than 3, or the total number of planar supports does not exceed the required percentage (i.e. 65% in our experiments). In the end, there are three categories of patches: those with reliable plane estimation under the plane fitting criterion (C_i=2) , those with unreliable plane estimation (C_i=1), and those without any plane estimation (C_i=0). At this point, each patch's representation can be updated as

$$\mathbf{R}_i = (\mathbf{c}_i, \mathbf{b}_i, \{\mathbf{R}_{ij}, j=1,...,J\}, \{\mathbf{p}_{il}, W_l, l=1,...,L\}, C_i = 0,1, \text{or } 2, \mathbf{\Theta}_i = (a_i, b_i, c_i, d_i)), i = 1,..., N \qquad (15)$$

The plane parameter set $\mathbf{\Theta}_i$ exists if $C_i \neq 0$. All the patches will go to the next stage for further processing.

Before we go to the next stage, we want to summarize the advantages of the patched-based natural matching primitives for stereo matching. First, treated separately, natural matching primitives on a patch represent the most salient visual features of the patch, and only contain pixels on that patch. Therefore, more accurate matches can be found for the patch that is textureless within and has a sharp depth boundary around. Second, taken together, more accurate and more robust results can be expected since these natural matching primitives are fitted on a single planar surface. Finally the algorithm is very efficient since only interest points of a region are matched in order to obtain the 3D of all the points within the region.

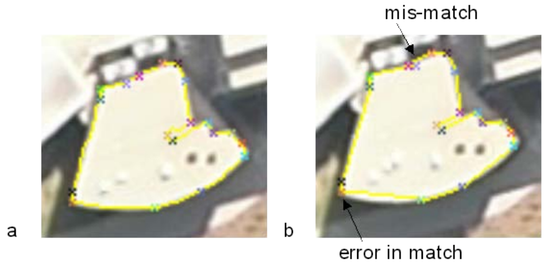

Fig. 7. An example of region matching results. The matches are marked as "X", with corresponding colors.

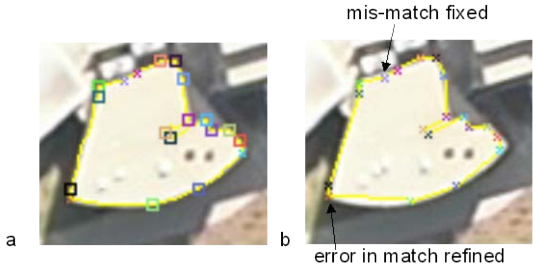

Fig. 8. An example of surface fitting results. Both the mismatch and the small error in the initial match are fixed.

Fig. 7 shows a real example of a natural-window-based stereo matching result for a static object (the roof of a building). The 19 interest points that are detected and their correspondences are marked on the boundaries in the left and right images, respectively. One mismatch and a small error in match are also indicated on the images. Fig. 8 shows the results of fitting and back-projection of the fitted region onto the right image. The 15 seed interest points (out of 19) used for planar fitting are indicated on the left image as squares. Both the mismatch and the small error in the initial match are fixed.

5.2. Refining plane parameters with multiple mosaics

After the above stereo matching is applied to the first pair of stereo mosaics, I_0 and I_1, initial estimations of the 3D structure of all the patches (regions) in the reference mosaic are obtained. Further matches between the reference mosaic I_0 and each of the rest of the mosaics, I_2, …, I_K, are then conducted. The initial visual displacement of each interest point on a patch is predicted from the result of this point estimated from the first stereo pair. From Eq. (2), we know the visual displacement Δy is proportional to the selected "disparity" (d_y) for a pair of stereo mosaics for any static point, i.e.,

$$\Delta y = (\frac{Z}{H} - 1) d_y \tag{16}$$

Therefore, the visual displacement of the interest point in consideration can be predicted except when the point is on a moving object, which will be reconsidered in the moving target detection stage. Assume that the visual displacement for an interest point is Δy_1 between I_0 and I_1, where $d_y = dy_0 - dy_1$, then between I_0 and I_k, where $d_y = dy_0 - dy_k$, the predicted visual displacement is

$$\Delta y_k = (\frac{d_{y0} - d_{yk}}{d_{y0} - d_{y1}})\Delta y_1 \tag{17}$$

For refining the initial estimates of visual displacements, the two-step algorithm in Section 5.1 is modified to obtain new plane parameters for each pair of stereo mosaics, with a very good initial estimation to start with to reduce the search range.

From the K pairs of stereo mosaics, up to K sets of plane parameters $\boldsymbol{\Theta}_{ik} = (a_{ik}, b_{ik}, c_{ik}, d_{ik}), k=1,\ldots,K$, are obtained for each region (patch) in the reference mosaic (some regions have fewer than K sets of available plane parameters due to the lack of sufficient numbers of interest points, or unreliable plane fitting). In order to obtain the most accurate plane parameters for each planar patch, the following steps are performed. First, for each pair of stereo mosaics, the patches in the reference mosaic are warped to the target mosaic in order to compute a color sum of square differences (SSD) for each region, between warped and original target images. Generalizing Eq. (1) to K views, and with 3D planar parameter estimation, we have

$$\begin{cases} x_k = F\frac{X}{Z} \\ y_k = F\frac{Y}{H} - (\frac{Z}{H} - 1)d_{yk} \\ a_k X + b_k Y + c_k Z = d_k \end{cases} \tag{18}$$

where the subscript i is dropped for simplifying the notations. Given a point $\boldsymbol{p}\ (x_0, y_0) \in \mathbf{R}_i$ in the reference view I_0, its 3D coordinates (X,Y,Z) can be calculated using Eq. (18), with k =0. Then again, using Eq. (18), the coordinates of the corresponding point in the *kth* view (k=1, 2,…, K), $\boldsymbol{p}_k\ (x_k, y_k)$, can also be obtained. We use a function Ψ_k to represent the above geometric transformation from the 0th view to the kth view:

$$\mathbf{p}_k = \Psi_k(\mathbf{p}) \tag{19}$$

Then the color SSD of the kth interest point of the region $\mathbf{R}_i$ can be calculated as

$$SSD_{ik} = \sum_{\mathbf{p}\in\mathbf{R}_i} |\mathbf{I}_k(\Psi_k(\mathbf{p})) - \mathbf{I}_0(\mathbf{p})|^2, k = 1,2,...,K \tag{20}$$

where $\mathbf{I}_0$ and $\mathbf{I}_k$ are the color vectors in the reference and the kth target views. Then, among all the estimates for each patch, the set of plane parameters with the least SSD value is selected as the best plane estimate. With multi-view refinements, the plane parameters and their categories in Eq. (15) are updated; some regions under the categories $C_i = 0$ or 1 may be upgraded into the category $C_i = 2$ under both the plane fitting criterion and multi-view refinement.

Note that using the knowledge of plane structure (i.e., 3D orientation), the best angle to view the region can be estimated, where the viewing direction of the selected mosaic (among all the possible viewing directions) is the closest to the plane norm direction. For example, for the side of a building that faces the right (refer to Fig. 2), the best match could be obtained from the first pair of stereo mosaics. If the view angle is equal to or greater than 90 degrees (relative to the plane norm), the region will not be visible. Incorporating this information, the SSD calculations are only carried out for those patches between the reference and target mosaics if the plane norms have less than 90-degree view angles from the viewing directions of the mosaics. Experimental results of improvements in 3D reconstruction will be shown in Section 7 with both real and simulated video sequences.

5.3. Plane updating using neighbors and global scene constraints

After the plane parameters with the smallest SSD value have been obtained for each region $\mathbf{R}_i$, we will have a close look at the best SSD of each region within category $C_i = 2$, under both the plane fitting criterion and multi-view refinement. If the SSD value is larger than a preset threshold T_i, then the patch is moved to *unreliable* category ($C_i = 1$) under plane fitting, multi-view refinement and SSD evaluation, therefore the attributes in Eq. (15) are further updated. Note that the SSD of the region $\mathbf{R}_i$ is calculated as the sum of all the pixels of 3 color components in the region, therefore T_i is defined as

$$T_i = Q_i \times 3 \times D^2$$

where is Q_i is the total number of pixels in the region $\mathbf{R}_i$, and D is the threshold of the difference between two corresponding components. In our experiments, we set D = 16 pixel levels of 512 possible differences. We have found that quite some small regions around a large region corresponding to a surface (or part) of a 3D object are generated by color segmentation, and are either marked as unreliable or without plane estimation. Therefore, we use two methods to update the plane parameter estimations: neighbor patch supporting and global scene constraints.

In the neighborhood supporting strategy, we perform a modified version of the neighboring plane parameter hypothesis algorithm (Tao, et al, 2001) to infer better plane estimates. Based on our region categorization, the main modification is that the parameters of a neighboring region are adopted only if it is marked reliable and the best neighboring plane parameters are accepted only when the match evaluation cost (SSD) using the parameters is less than the threshold T_i for the ith region $\mathbf{R}_i$. Our neighbor supporting algorithm has the following steps.

(1). Select reliable regions $\{\mathbf{R}_{i,j1}, \mathbf{R}_{i,j2},\ldots, \mathbf{R}_{i,jM}\}$from the set of neighboring regions $\{\mathbf{R}_{ij}, j=1,2,\ldots J\}$ for the current region $\mathbf{R}_i$, including the current region, therefore M<= J+1.
(2). Apply the parameter set $\mathbf{\Theta}_{jm}$ (m=1, 2, …M) to the region $\mathbf{R}_i$, to calculate the corresponding $SSD_{i,jm}$(m=1, 2, …M), using Eq. (20).
(3). Select the parameter set $\mathbf{\Theta}_{jm}$(1 <= m <= M) that gives the smallest SSD, for the current region.

With the neighborhood supporting, a un-estimated ($C_i = 0$) or un-reliable region ($C_i = 1$) can be upgraded to a reliable region (with $C_i = 2$) if its best SSD is smaller than the threshold T_i; the plane parameters of most of the regions can be refined no matter what categories they initially were. Further, if the neighboring regions sharing the same plane parameters, then they are then merged into one reliable region. This step is performed recursively till no more merges occur. We set a tight threshold so that only those very static regions with very reliable matches are classified into category 2. The motivation is to have most of the regions for moving targets labeled into un-estimated or un-reliable regions for further investigation. In doing this, some of the static regions will be classified into category 0 or 1, but there are typically small regions along the boundaries of buildings that can be further discarded as noise.

We have also explored global scene constraints to improve the robustness of 3D reconstruction for highly cluttered urban scenes, where a lot of small patches are generated. In a typical urban scene, many surfaces such as facades, rooftops, roads, etc., share the same plane directions. Therefore, in applying the global scene constraints, after an initial pass of plane parameter estimation with multiple views, the top several dominant plane directions are obtained by a simple clustering algorithm on those reliable regions. Then the following two steps are performed.

(1) For those regions that either are marked as unreliable (due to plane fitting or SSD evaluation), or do not obtain sufficient good local matches (L<3), the parameters of the dominant planes can be used to guide the search and the refinement of their matching and plane fitting steps. Since each plane only has 4 parameters (a, b, c and d), and the norm of each dominant plane provide 3 of them (i.e., a, b and c), the rest of the job is simply to compute the variable d. Therefore, for each region with at least one reliable local match among the detected interest points, we plug this reliable match into the plane equation using each of these domination plane norms, to obtain possible estimations of d. Then, we compute the SSD of the corresponding patch pair (i.e. the warped reference patch and the original target patch) based on each estimate of the parameter d, and finally select the one with the smallest SSD score as the result.

(2) After applying the global scene constraints, neighborhood hypothesis (as discussed above) is applied to *all* the regions to generate more reliable and accurate 3D estimation results.

Experimental results on plane merging and local/global scene constraints will be shown in Section 7, with both simulated and real video sequences.

5.4. Moving object detection

After the plane merging stage, most of the small regions are merged together and marked as reliable. Moving object patches that move along epipolar lines should also obtain reliable matches after the plane merging step, but they appear to be “floating” in air or below the surrounding ground, with depth discontinuities all around it. In other words, they can be identified by checking their 3D anomalies (Section 3.2, observation (4)). This is mostly true for aerial video sequences, where ground vehicles and humans move on the ground. For ground video sequences, the multiple mosaic approach discussed in Section 4 can be applied. This remains our future work.

In general cases, most of the moving targets are not exactly on the direction of the camera’s motion, therefore, those regions should have been marked as unreliable in the previous steps. Regions with unreliable matches fall into the following two categories: (1) moving objects with motion not obeying the pushbroom epipolar geometry; (2) occluded or partially occluded regions, or regions with large illumination changes. For regions in the second category, their SSDs in stereo matching evaluation are always very high. The regions in the first category correspond to those moving objects that do not move in the direction of camera motion; therefore they do not obey the pushbroom stereo epipolar geometry. Therefore, for each of these regions, we perform a 2D-range search within its neighborhood area. If a good match (i.e., with a small SSD) is found within the 2D search range, then the region is marked as a *moving* object. We can also take advantage of the known road directions, to more effectively and more reliably search for matches of those moving vehicles. The road directions can be derived from 3D reconstruction results, e.g., in a city scene, the norm directions of the two dominant planes of the building façades surrounding the ground area on which the moving objects reside.

In the current implementation of moving target detection (ground vehicles) from aerial images, large occluded regions are still not well processed and consequently confuse the moving target detection as described above. Therefore, the size of each region is also taken into account to classify it as a moving target. The typical size (Dx*Dy) of a vehicle can be roughly estimated based on its physical size (ΔX, ΔY), the image resolution of images (characterized by the focal length f in pixels), and the altitude Z the camera from the ground:

$$(Dx, Dy) = (F\frac{\Delta X}{Z}, F\frac{\Delta Y}{Z}) \qquad (21)$$

With the settings of the image sequences in our experiments, most of the moving vehicles are less than Dx*Dy = 300 pixels in images.

The moving target detection steps are summarized as follows.

(1) For all reliable regions with less than Dx*Dy pixels, the 3D anomaly condition is checked. If one of the following conditions is satisfied, then a region $\mathbf{R}_i$ goes through 2D region search to find its motion parameters (S_x, S_y), and is marked as a moving target if the SSD is smaller than the preset threshold T_i: (a) the height of the region $\mathbf{R}_i$ is 20 meters higher than the average height of the neighboring regions $\{\mathbf{R}_{ij}\}$; or (2) the height of the region $\mathbf{R}_i$ is 10 meters lower than the average height of the neighboring regions.

(2) For *all* unreliable regions with less than Dx*Dy pixels, the epipolar constraint is applied. Each region $\mathbf{R}_i$ in this class goes through 2D neighborhood search to find its motion parameters (S_x, S_y), and is marked as a moving target if the SSD is smaller than the preset threshold T_i.

At the end of all the four stages, a region $\mathbf{R}_i$ is represented as the following form:

$$\mathbf{R}_i = (\mathbf{c}_i, \mathbf{b}_i, \{\mathbf{R}_{ij}, j = 1,..., J_i\}, C_i, \mathbf{\Theta}_i = (a_i, b_i, c_i, d_i), \mathbf{m}_i = (S_{xi}, S_{yi})), i = 1,..., N \qquad (22)$$

where C_i is redefined as reliable static region (C_i = 2), moving target (C_i=1), and unreliable region (C_i=0), $\mathbf{m}_i$ is the motion vector if the region is a moving target. Note that we have removed the interest points and natural matching primitives from each region in Eq. (22), which are only used during the 3D estimation process. And more precisely, the number of neighboring region for the region $\mathbf{R}_i$ is noted as J_i (i=0,…, N).

6. CB3M: Content-Based 3D Mosaics

The output of the two-phase processing – pushbroom mosaicing and content extraction, is a content-based 3D mosaic (CB3M) representation. It is a highly compressed visual representation for very long video sequences of a dynamic 3D scene. In the CB3M representation, the panoramic mosaics are segmented into planar regions, which are the primitives for content representation. Each region is represented by its mean color, region boundary, plane normal / distance, and motion direction / speed if it is a dynamic object. Relations of each region with its neighbors are also built for further object representations (such as buildings, road networks) and automatic target recognition.

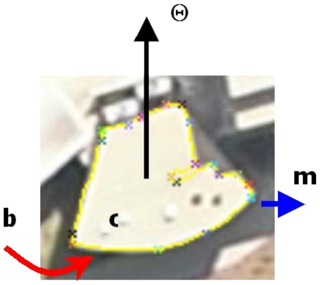

Fig. 9. Content-based 3D mosaic representation

6.1. Basic content-based 3D mosaic representation

In our current basic implementation, a content-based 3D mosaic (CB3M) representation is a set of video object (VO) primitives (i.e., patches, e.g. in Fig. 9) that are defined as

$$\mathbf{CB3M} = \{\mathbf{R}_i, i = 1, \ldots, N\} \tag{23}$$

where Ri is defined in Eq. (22). As a summary, they are explained below:

(1) N is the number of VOs, i.e., “homogeneous” color patches (regions);

(2) $\mathbf{c}_i$ is the color (3 bytes) of the *ith* region;

(3) $\mathbf{b_i}$ is the 2D boundary of the *ith* region in the left mosaic, chain-coded as $\mathbf{b_i} = \{(x_0, y_0), G_i, b_1, b_2, \ldots b_{Gi}\}$, where the starting point (x_0,y_0) has 4 bytes, and each chain code has 3 bits. G_i is the number of boundary points (which needs 4 bytes each) and $G = \sum G_i$ is the total for all regions;

(4) $\{\mathbf{R}_{ij}, j = 1, \ldots, J_i\}$ is the list of the labels of neighboring regions of the ith region, each needs 4 bytes (assuming on average the number of neighboring regions for each region is J, i.e. $J = (1/N) \sum J_i$);

(5) $C_i = 2$, if the region is a static patch with reliable plane parameters (see (6)); $C_i = 1$, if the region is a moving target (therefore with mi, see (7)); $C_i = 0$, otherwise (unreliable, maybe occluded regions).

(6) $\mathbf{\Theta_i} = (a_i, b_i, c_i, d_i)$ represents the plane parameters of the region in 3D, 4 bytes for each parameter; and

(7) $\mathbf{m_i}$ represents the M motion parameters of the region if in motion (e.g. M =2 for 2D translation (S_x, S_y) on the ground).

Therefore the total data amount is (without counting C_i)

$$N_{color} + N_{boundary} + N_{neighbor} + N_{structure} + N_{motion}$$
$$= 3N + (8N+3G/8) + 4JN + 4*4N+4M*N_m$$

$= (27+4J)N+3G/8+4MN_m$ (bytes) (24)

when each of the motion and structure parameters needs 4 bytes. In the above equation, N_m is the number of moving regions (which is much smaller than the total region number N). Note that the VO primitives are those patches before region merging in order to preserve the color information.

The proposed CB3M representations are highly compressed visual representations for very long video sequences of dynamic 3D scenes. The representations could fit into the MPEG-4 standard (Koenen, et al, 1997), in which a scene is described as a composition of several Video Objects (VOs), encoded separately.

The CB3M construction and representation provides the following benefits for many applications, such as urban transportation planning, aerial surveillance, robot navigation and urban modeling. A long image sequence of a scene from a fly-through or drive-through is transformed in near real time into a few large FOV *panoramic mosaics*. This provides a synopsis of the scene with all the 3D objects and dynamic objects in a single view. The *3D contents* of the CB3M representation provide three-dimensional measurements of objects in the scene. Since each object (e.g. a building) has been represented into 3D planar regions and their relations, further object recognition and higher-level feature extraction are made possible. The *motion contents* of the CB3M representation provide dynamic measurements of moving targets in the scene. For example, in traffic monitoring, the motion and 3D contents not only provide information about the vehicles' directions and speeds, but also the traffic situation of a road segment since each road "region" can also be extracted based on its 3D information and shape, and the statistics of the moving objects on the road can provide very useful traffic information. Finally, the CB3M representation is *highly compressed*. Usually a compression ratio of thousands to ten thousands can be achieved. This saves space when a lot of data for a large area need to be archived. Real examples of coding and compression will be provided in the following section.

6.2. Discussions: occlusion representation and higher level object representation

Since the basic CB3M representation is a set of planar patches with shape and appearance properties, it can be naturally extended to represent relations between regions, and occluded regions that are not visible or only partially visible in a single reference mosaic used as the base image of the basic CB3M representation. In the current implementation, only 3D parametric information of planar patches in the reference mosaic is obtained. Since different visibilities are shown in mosaics with different viewing directions, we want to extend the approach presented in Section 5 to produce multiple depth maps with multiple reference mosaics and then integrate the results by performing occlusion analysis. The neighboring regions of each patch have been extracted in the patch and interest point extraction step. This lays a solid foundation for object recognition and occlusion handling, which will be our future work. Then an extended content-based 3D mosaic representation can be generated by inserting the occluded regions in the basic CB3D representation, similar to the layered representation we have proposed in Zhu and Hanson (2004). In the end, the extended CB3M representation will have the following three components:

(1) A base layer that consists of a set of planar patches corresponding to the reference mosaic;
(2) A set of occluded patches that are not visible in the reference mosaic, but are visible in other views, together with the corresponding viewing direction information for these patches; and
(3) All the neighboring regions of each patch, including the base patches and occluded patches.

With these three components, and the corresponding viewing direction information, the extended content-based 3D mosaic representation can be easily converted into other representations, such as digital elevation map, and be used for image-based rendering since both the shape/appearance information and the viewing information are available. Furthermore, developing higher-level representations that group the lower-level natural patches into objects (vehicles, buildings, roads, humans) are also possible, for applications such as automated target recognition and 3D model indexing.

7. Experimental Results and analysis

The proposed approach for the content-based 3D mosaic representations was applied to multi-view pushbroom mosaics generated from real world video sequences. Here we present three examples: the flyover of a campus scene, a ground vehicle drive-by in an indoor scene, the flyover of a New York City (NYC) scene, and the results on AFRL Columbus Large Format Image (CLIF) dataset. We also performed evaluations on the accuracy of 3D and motion estimation on a simulated video sequence generated with the ground truth data. Finally we will provide some analysis on computation time in both stereo mosaicing and content extraction.

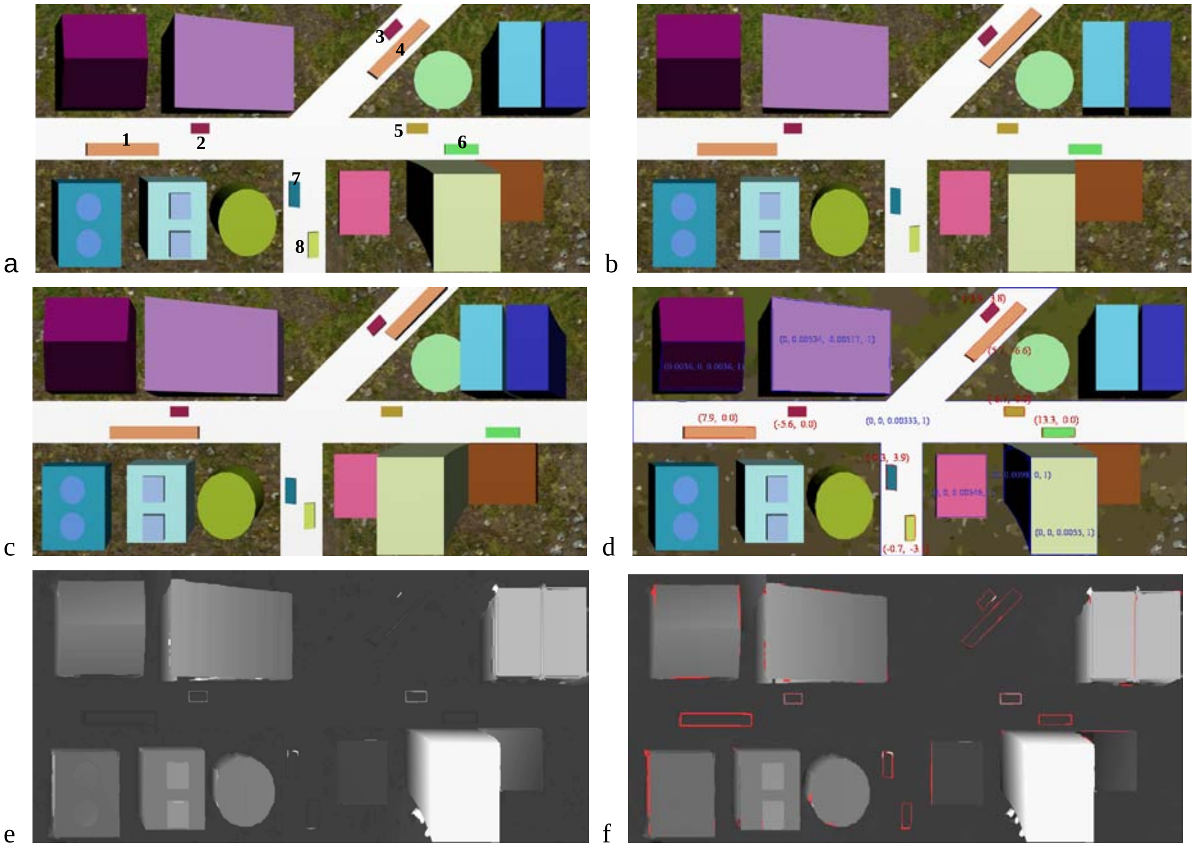

Fig 10. (a) The leftmost, (b) center and (c) rightmost views of the nine mosaics of a simulated scene. The final CB3M representation is shown in (d). Each region is rendered by its average color. Plane parameters *(a,b,c,d)* (in blue) and boundaries for several representative surfaces, and motion displacements (s_x, s_y) (in red) of the detected moving targets are labeled in (d). For comparison, (e) and (f) show the rendered "height" maps of the scene from the stereo matching results from the 1st stereo pair only, and from all the mosaics, respectively. Finer and more accurate results are obtained in (f). Regions marked in red are the "outliers" that will be passed to the moving target test; some of them are due to occlusions at depth boundaries rather than independent motion, but they are too thin or too small to be real moving targets. The detected moving targets are shown in (d).

7.1. Results and analysis on a simulated scene

Nine parallel-perspective stereo mosaics were generated from a simulated video sequence of a simulated scene with ground truth data of both 3D and moving targets (Fig. 10). The sequence was generated using the following parameters. The virtual "aircraft" with a video camera flew at a 300-meter height above the ground along a 1D translational direction, and the motion direction is perpendicular to the optical axis of the camera. The focal length of the camera is 3000 pixels (as in Eqs. (1), (4) and (6)), and the camera moves with a constant speed. The 3D "buildings" are with heights from 5 to 120 meters above the ground, with different roof shapes (rectangular, round, frontal, ridged, slanting, and/or with small attachments). There are occlusions between buildings. Each of the eight moving objects has a height from 2 to 5 meters, and undertakes a 2D translational motion with constant velocity during the period of the capture of the total 1640 frames of images, except the one labeled as "1" in Fig 10a, which varies in velocity. The velocity of the motion of each moving target is represented in centimeter (cm) per frame. Nine 1-column width slit windows are used to generate the nine mosaics (refer to Fig. 4), every pair of the two consecutive windows has a 40-pixel distance, and hence the total distance between the first and the last slit windows is 320 pixels. Fig. 10 only shows three of the nine mosaics, (a) the leftmost, (b) the center, and (c) the rightmost views. Varying occlusions/visibilities can be seen in these mosaics. The change of velocity of the 1st moving target can be seen from the varying sizes of its images in the three mosaics.

From the nine mosaics, we use the leftmost mosaic as the reference image to match with the other eight mosaics. For each region in the reference mosaic, there are 8 plane estimation results, and the best estimate is selected for the 3D parametric representation of the region. The final "height" map (Fig. 10f) is rendered as a map of heights of objects from the ground, i.e. $-H\,\Delta y/d_y$, (normalized to a range from 0 to 255 for display). For comparison, we have also generated a height map (Fig. 10e) from the stereo matching results of only the first and the second mosaics (without region merging). It can be seen that by using the best parameter selections from multi-view mosaics and utilizing the plane merging step, finer 3D results are obtained for many building roofs, and more accurate results are obtained for sides of buildings.

We have also compared the final estimated height map with the ground truth data. The error histogram (base 2 logarithmic scaling on the number of pixels) is shown in Fig. 11a for all the regions (including the moving object regions and other obvious wrong matches). From the error distribution, we have found that the errors of 86.5% points in the reference mosaic are within ±4 meters. The absolute average value of the errors for those points is only 0.317 meters. Note that in theory, the error of the depth/height estimation by the pushbroom stereo in Eq. 2 can be calculated as $\delta Z = (H/d_y)\ \delta y$, where δy is the error in stereo matching (in pixels). In this experiment, H is 300 meters, and d_y is from 40 to 320 pixels (from the first pair to the 8th pair of stereo mosaics), and ideally δy is 0.1 pixels with the sub-pixel local match step. Therefore, the theoretical errors after local match go from 0.75 down to about 0.1 meters from the first pair to the 8th pair. However, larger viewing differences introduce larger errors in δy, therefore the error reduction by using larger disparities (from 40 to 320) is not as significant as the theoretical estimation. On the other hand, plane fitting on the multiple interest points with sub-pixel accuracy increases the accuracy in δZ, which leads to a more realistic error range close to the average error of the estimated depths/heights in this experiment (i.e., 0.317m). To show how depth errors vary and how the planar parameters are selected among the eight pairs of stereo mosaics in generating the final height map, Fig. 11b shows the estimation errors of the planar parameters (from the ground truth) for the 17 largest regions in the reference mosaic. Most of the depth errors are below 0.3 meters, and the magnitudes are comparable among different pairs of stereo mosaics with various "disparities" (i.e., *dy*). Because of this reason, for each region, we select the "best" result of plane parameter estimation among the eight stereo pairs, instead of using the last pair with the "largest" disparities. Nevertheless, the multi-view approach outperforms the two-view approach significantly. Table 1 compares the average depth errors of depth estimations from a pair of stereo mosaics and all 8 pairs of mosaics for all pixels, 85% of pixels and 75% pixels, respectively. It clearly indicates that multi-view approach reduces the depth errors to about 50%.

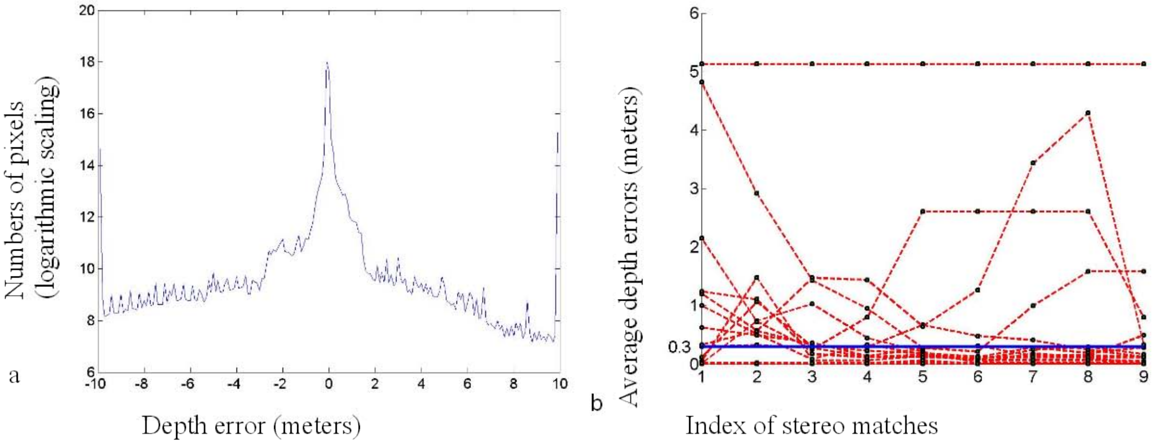

Fig 11. Depth error analysis. (a) Error histogram. (b) Comparison and selection among the results from the 8 pairs of stereo mosaics for the largest 17 regions. The last column (9th) shows the final selection.

Table 1. Comparison of average depth estimation errors: two views and multiple views

# pixels	75%	85%	100%
$e_{1st\text{-}pair}$	0.20m	0.54m	5.42
$e_{all\text{-}pairs}$	0.07m	0.20m	3.65

After the regions have been merged, we analyze all the reliable regions, and those with obvious 3D anomalies are marked as moving objects traveling along the epipolar lines. For example, in Fig. 10a, the heights of the regions labeled 1 and 6, if treated as static objects, are estimated as -39 meters and -50 meters "high" from the ground, respectively, much lower than the ground plane. The small regions labeled 2 and 5 are estimated as 94 meters and 98 meters high from the ground, respectively, much higher than the ground. In fact all these regions only are 2 to 5 meters high from the ground. So these regions with such 3-D "anomalies" if incorrectly treated as static objects are detected as moving targets.

Table 2. Motion estimation errors

Obj Idx	Ground Truth (cm/frame)		Estimated Results (cm/frame)		Errors (cm/frame)	
	Sx	Sy	Sx*	Sy*	ΔSx	ΔSy
1	0	2.485	0	1.649	0	0.836
2	0	-1.499	0	-1.628	0	0.129
3	1.064	-1.262	1.053	-1.08	0.011	-0.181
4	-1.414	1.414	-1.444	1.247	0.031	0.166
5	0	-1.999	0	-2.012	0	0.013
6	0	2.499	0	2.495	0	0.003
7	0.999	0	0.982	-0.076	0.017	0.076
8	-0.781	0	-0.789	-0.178	0.007	0.178

On the other hand, those unreliable regions (as possible candidates for moving objects not along the epipolar lines) further go through 2D-range searches for matches within their neighborhood areas (e.g., 30x30-pixels 2D range). In

Fig. 10a, regions 3, 4, 7 and 8 are moving targets. They do not obtain reliable matches in the stereo match step, but could find reliable matches from their 2D range searches, between the first mosaic and the rest of the mosaics. Therefore they are considered as moving targets. Note that those regions marked with red boundaries in the height map have good matches in their 2-D range searches; however, most of them are (1) just at the depth boundaries of dramatic depth changes, and (2) have very small sizes, or have very thin structures, therefore are not considered to be moving targets by using these two criteria. (Same treatment is done for motion detection in real video experiments below.) The estimated motion parameters ($s_{x,},s_y$) (in pixels) of those detected moving targets from the first pair of stereo mosaics are marked on the CB3M map in Fig. 10d. The error analysis results of the 8 detected moving targets are shown in Table 2. The average error of the 2D motion estimation is (0.198, 0.008) in velocity (cm/frame), or (0.791, 0.033) in displacements (pixels) between the first pair of the stereo mosaics. The error for the 1st object is the largest since its velocity is not constant.

The compression of a video sequence comes from two steps: stereo mosaicing and then content extraction. For the simulated image sequence, we have 1640 frames of 640*480 color images, so the data amount is 1.44 GB. The size of pair of the stereo mosaics is 1320*640*2, which has 4.83MB (without compression). The two mosaics in high-quality JPEG format only have 2*75 KB; therefore, a compression ratio of about 9,837 is achieved for the stereo mosaics (the first step). If all the nine mosaics are saved for mosaic-based rendering (Zhu & Hanson, 2006), the data amount will be 9*75KB hence the compression ratio is about 2,186.

Then after color segmentation, 3D planar fitting and motion estimation, we obtained the CB3M representation (Fig. 10d) of the video sequence, with the total number of the natural regions N = 1,342 and the total number of boundary points G = 119,477. The total amount of data in its CB3M representation is 80.8 KB (with a header). This real file size is consistent with the estimation of data amount using Eq. (24), which is about 79.2 KB (without the coding of the information of neighboring regions for each region; same below). The data amount is reduced to 19.4 KB with a simple lossless WinZip compression on the CB3M data; therefore, the compression ratio is about **76,061:1**. Note that the compression ratio depends on how fine the color segmentation is. In the example shown in Fig. 10d, the main visual features of the scene are coded. More importantly, the CB3M representation has object contents which can be used for object indexing, retrieval and image-based rendering. The plane parameters (a,b,c,d) for the several representative regions are shown on the CB3M map in Fig. 10d (from left to right: one side of a ridged roof, a slanting roof, ground with depth Z= 300.0m, roof of a low building with Z = 289.0m, and side and roof of a tall building with Z=180.0 m), all measured from a camera 300 meters above the ground.

7.2. Results on real video data: a campus scene

The first real video sequence we tested our approach on is for a campus scene captured by a camera on a light airplane flying about 300 meters above the ground. The camera was calibrated using some ground truth data. The image resolution is 640*480. Nine mosaics were generated from the 1000-frame aerial video. Fig. 12a shows a pair of stereo mosaics (embedded in red and green-blue channels, respectively) from the nine mosaics, and two close-up windows are marked in the stereo mosaics, which include both various 3D structures and moving objects (vehicles). Fig. 12b is the "height" map (corresponding to the reference mosaic) using the proposed method. Fig. 12c and Fig.12d, Fig. 12e and Fig.12f show the images of the two close-up windows and the corresponding "height" maps. Note the sharp depth boundaries are obtained for the buildings with different heights and various roof shapes. The average heights of the buildings marked as A, B, C, D and E in Fig. 12d and Fig. 12f are 11.5m, 5.8m, 5.4m, 14.9m and 7.8m, respectively. The long building (D) has a slanting roof (left side is higher). Even though we have not conducted an accurate evaluation due to the lack of ground truth data, these estimations are consistent with the real heights of these buildings. The moving objects that have been detected across all the nine mosaics are shown by their boundaries (in red). Those vehicles that are not detected by our algorithm are marked by rectangular bounding boxes; they are either stationary (as those in the boxes 2 and 3), or deformed differently across the mosaics due to the changes of motion in velocities (as in the box 1) and directions (as in the box 4).

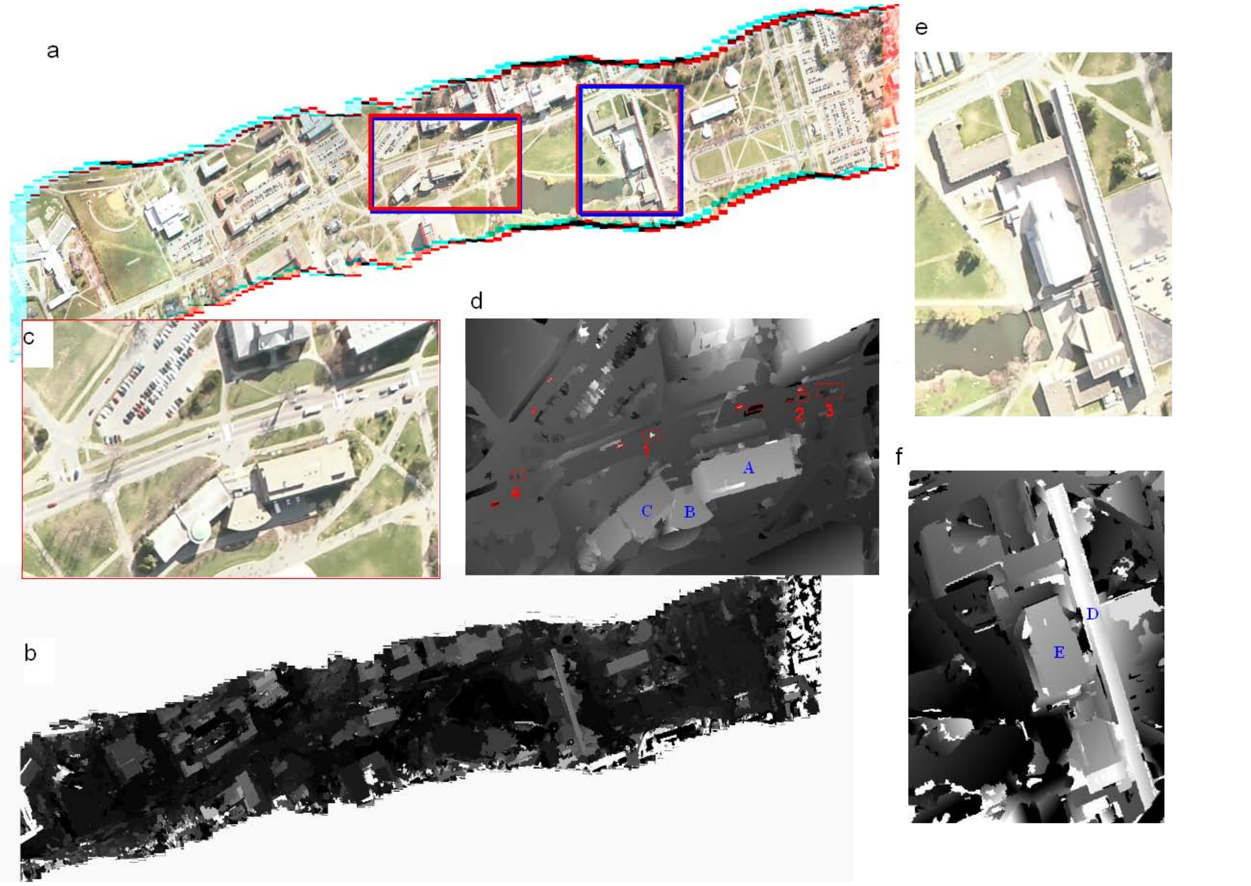

Fig. 12. 3D and motion from multi-view stereo mosaics of an aerial video sequence. (a) A pair of stereo mosaics from the total nine mosaics; (b) height map of entire mosaic; (c) close-up of the 1st window marked in (a); and (d) the height map of the objects inside that window, with the detected moving targets marked by their boundaries and those not detected by rectangular boxes; (d) close-up of the 2nd window marked in (a); and (f) the height map of that window.

The CB3M mosaic (of the first window in Fig. 12a) is shown in Fig. 13, with a color, a boundary, plane parameters and a motion vector (if in motion) for each patch (region). Again we examine the compression of the real video sequence from two steps: stereo mosaicing and then content extraction. For the real image sequence, we have 1000 frames of 640*480 color images, so the data amount is 879 MB. The size of pair of the stereo mosaics (Fig. 12a) is 4448*1616*2, which has 41MB (without compression and with more than half empty space due to the fact that the mosaics go in a diagonal direction). The two mosaics in high-quality JPEG format only have 2*560 KB; therefore, a compression ratio of about 800 is achieved for the stereo mosaics (the first step). If all the nine mosaics are saved for mosaic-based rendering, then the data amount is 9*560KB so the compression ratio will be 179.

Then after color segmentation, 3D planar fitting and motion estimation, we obtained the CB3M representation of the video sequence, with the total number of the natural regions N = 6,112 and the total number of boundary points G =

420,445. The total amount of data in its CB3M representation is 316 KB (with a header). This real file size is consistent with the estimation of data amount using Eq. (24), which is about 315 KB. The data amount is reduced to 90 KB with a simple lossless WinZip on the CB3M data; therefore, the compression ratio is about **10,001**. Note that the CB3M representation in Fig. 13 consists of regions corresponding to rather large object surfaces in order to rapidly obtain robust 3D structures. However, fine details are not preserved. In our previous experiments, we over-segmented the reference mosaic so that finer details of the scene can be coded. In that case the compression ratio was still over 2000.

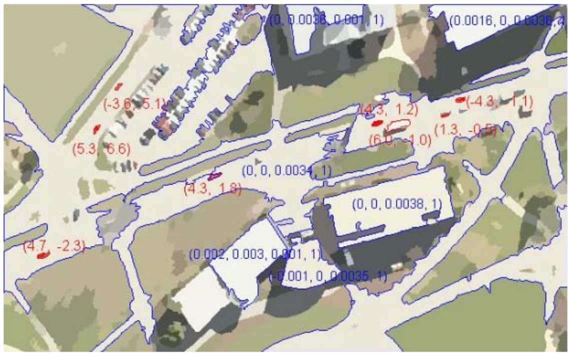

Fig. 13. Content-based 3D mosaic representation of an aerial video sequence. Only a window is shown, with some of the regions labeled by their boundaries and plane parameters (in blue), and the detected moving targets marked by their boundaries and motion vectors (in red).

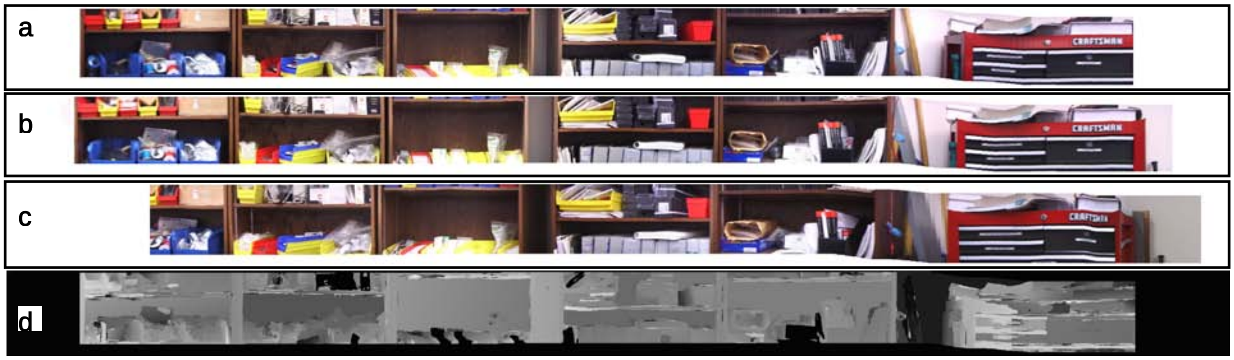

Fig. 14. Multi-view stereo mosaics for an indoor scene. (a) The leftmost, (b) center and (c) the rightmost views of total eleven mosaics. (d) the height map generated.

7.3. Results on real video data: an indoor scene

The second group of real-scene mosaics (Fig. 14) was generated from a video sequence of an indoor scene, the side view of several bookshelves and a file cabinet against a wall, captured by a video camera mounted on a ground robot. The video sequence has 1020 frames of 320*240 color images. We show this example to indicate that the same approach applies to ground video sequences and indoor scenes. Eleven mosaics were generated and used as input data for our algorithm to generate a height map of the entire scene. Three mosaics and the final "height" map are shown in Fig. 14, with the "height" values measured from the reference plane H. Note that height values are also obtained for textureless regions and thin structures. This information can be used for localization of a mobile robot along a long route by looking at the side of the route.

7.4. Results on real video data: an NYC scene

The NYC mosaics were generated from a video sequence from an NYC HD (high-definition) aerial video dataset (vol. 2) we ordered from http://www.artbeats.com/prod/browse.php. The video clip, NYC125H2, has about 25 seconds, or 758 frames of high-definition progressive video (1080*2000). Rooftops and city streets are seen as the camera looks ahead and down in a close flight just over One Penn Plaza and beyond in New York City. Yellow taxicabs make up a noticeable percentage of the vehicles traveling the grid of streets in this district of mostly lower-rising buildings, but have a few high-rise buildings. You may view the low-resolution version of the video following the link we have provided above. Our main task is to recover the full 3D model of the area automatically, with cluttered buildings with various heights, from less than ten to more than a hundred meters. Fig. 15 shows one of the four multi-view mosaics generated and used for 3D reconstruction and moving target detection. The mosaic that is shown here has been turned 90 degrees, therefore the camera moves in the direction from the left to the right in the mosaic. The size of the mosaic is 4816 (W) x 2016 (H). The camera slightly tilted to the up-right side so the ground plane in the mosaic is not leveled. You can clearly see this effect in the depth maps in Fig. 16.

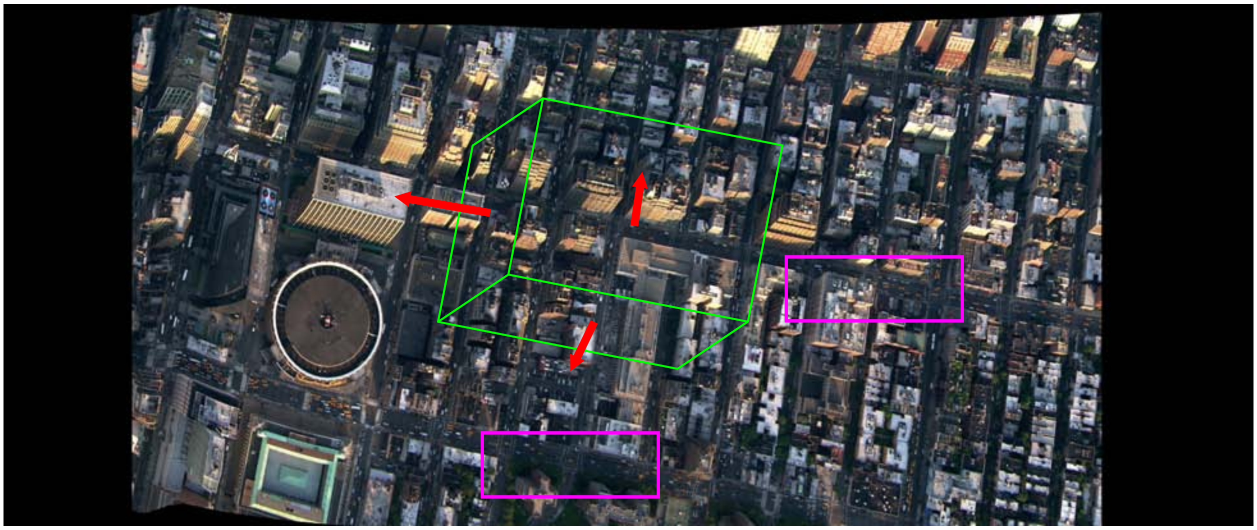

Fig. 15. A 4816 (W) x 2016 (H) mosaic from a 758-frame high-resolution NYC video sequence. The Manhattan world geometric constraint is illustrated on the mosaic.

This data set is very challenging due to the cluttered buildings and complex micro-surface structures that produce a lot of small homogeneous color patches after color segmentation. The regions with low-rising buildings (the right-hand side of the mosaic) do not have salient visual features and sufficient disparity for reliable depth estimation. So in this example, we also applied the Manhattan world geometric constraint (Coughlan & Yuille, 1999) to further refine the 3D reconstruction results. The detailed discussion of our algorithm is presented in our previous paper (Tang &Zhu, 2008b). As shown in Fig. 15, most of the planes (roads, rooftops and facades of buildings) are either perpendicular or parallel to each other, therefore, they consist of three orthogonal domination plane directions. In our experiments, among of all regions that have successfully obtained plane-fitting results from multi-view mosaics, those with reliable matches are used to automatically vote for the three domination planes. The three plane norms are [5.544, 1.360, 1.000], [-0.792, 3.837, 1.000] and [-0.026, -0.318, 1.000]. A simply cross-product check verifies they are almost orthogonal to each other (The angles between them are 85.52^o, 86.03^o and 92.69^o). The information of these three domination plane directions is very useful in both refining the 3D reconstruction and extracting moving targets.

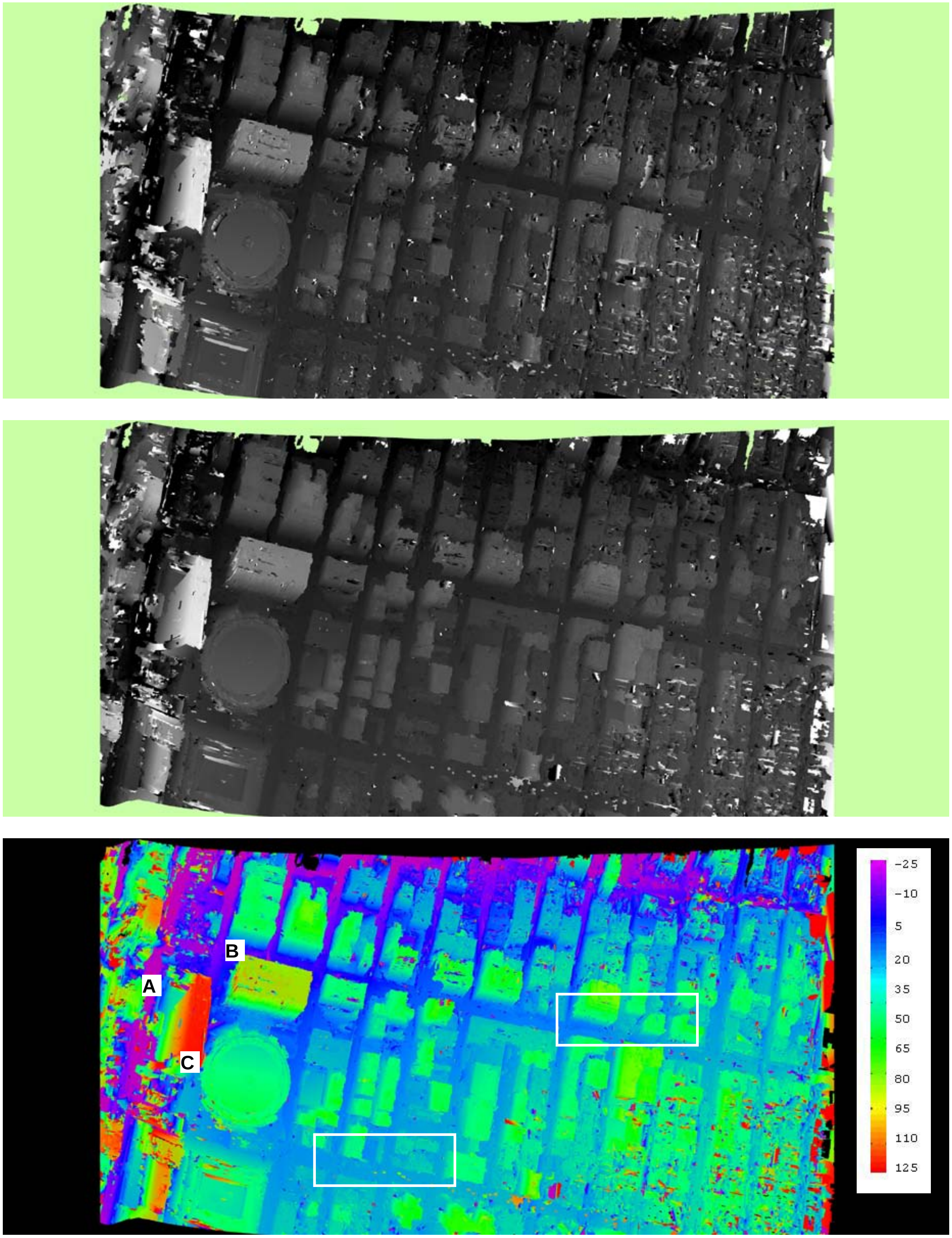

Fig. 16. (a) Depth from a pair of mosaics, (b) from four mosaics, and (c) color-coded depth map of (b)

Then, the rest of regions, i.e., the "outliers", go through the moving object detection test. We use the same method as presented in Section 5.4, and for this NYC data, we take advantage of the known road directions, to more effectively and more reliably search for matches of those moving vehicles. The road directions are derived from the two dominant planes of the building façades (the third one is for the ground and rooftop).

Fig. 16 shows the 3D reconstruction results of the NYC video data, all represented in the leftmost mosaic - the reference view. In Fig. 16a, the height map is rendered from the 3D structure result reconstructed from the first pair of stereo mosaics. It can be seen that the right-hand side has many spurious small regions. Fig. 16b shows the height map rendered from the result from the integration of the 3 stereo pairs of the four mosaics. It is obvious that the height map has improved significantly. The height map looks much smoother; many spurious depth estimations and small regions without reliable estimations are filled. Fig. 16c shows the colored coded height map from multi-view mosaics (same as Fig. 16b). The color bar on the right-hand side shows the correspondences of colors and height values. Due to the lack of the flight and the camera parameters, we roughly estimate the main parameters of the camera (i.e., the height H and the focal length F) from some known buildings. However, this gives us a good indication of how well we can obtain the 3D structure of this very complex scene. For example, the average heights of the three buildings at One Penn Plaza (marked as A, B and C in Fig. 16c) are 105.32 m, 48.83 m, and 19.93 m, respectively. Our approach handles scenes with dramatically varying depths. Readers may visually check the heights of those buildings with GoogleEarth. Note that the camera was not pointing perpendicularly down to the ground and therefore the reconstructed ground is tilted. This can be seen from the colors of the ground plane.

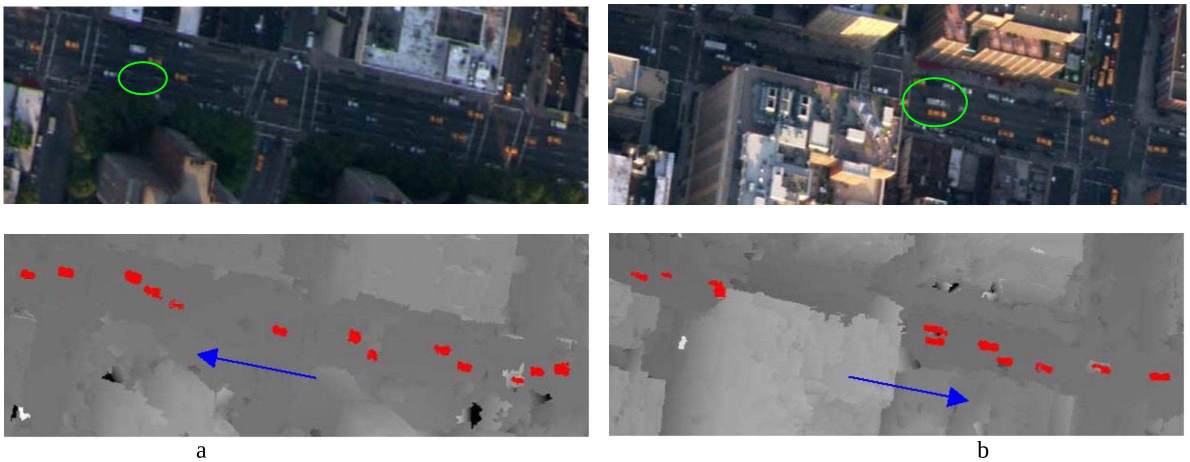

Fig. 17. Moving target detection using the road direction constraint. In the figure (a) and (b) are the corresponding color images and height maps of the 1st (bottom-left) and 2nd (top-right) windows in Fig. 15, with the detected moving targets painted in red. The two circles show the three moving targets that are not detected. The arrows indicate the directions of the roads along which the moving targets are searched.

The moving objects (vehicles) create "outliers" in the height map, as can be clearly seen on the color-coded map. For example, on the one-way road indicated in the first window in Fig. 15, vehicles moved from the right to the left in the figure, therefore, their color-encoded "height values have more red/yellow colors (i.e., the estimated heights are much higher than the ground if assumed static). On the other hand, on the one-way road indicated in the second window in Fig. 15, vehicles moved from the left to the right in the figure, therefore, their color-encoded "height values have more blue colors (i.e., the estimated heights are much lower than the ground if assumed static). After further applying the constraint of road directions that we obtained from the dominant plane voting, moving targets are searched and extracted. In Fig. 17, all of the *moving* targets (vehicles) are extracted, except the three circled in the figure. These three vehicles are merged with the road in color segmentation. Other vehicles that are not detected were stationary; most of them are on the orthogonal roads with red traffic signals on for stop, and a few parked on these two one-way roads.

7.5. Code modification and results on AFRL CLIF data

We also made modifications to the mosaicing software. The GUI interface was removed since it was Windows specific. In addition we are now using the OpenCV library to handle image IO. This allows the program to use all major supported image formats (jpeg, png, bmp, etc). These changes have now made the software cross-platform, it will compile on Windows (Visual C++), and Mac OS X/UNIX (g++). We are also making use of SIFT features for the ray interpolation in mosaic generation. Using the new software system, we have generated mosaics from both EO and IR images of the CLIF 2006 datasets, and preliminary 3D reconstruction results have also been obtained (Fig. 17).

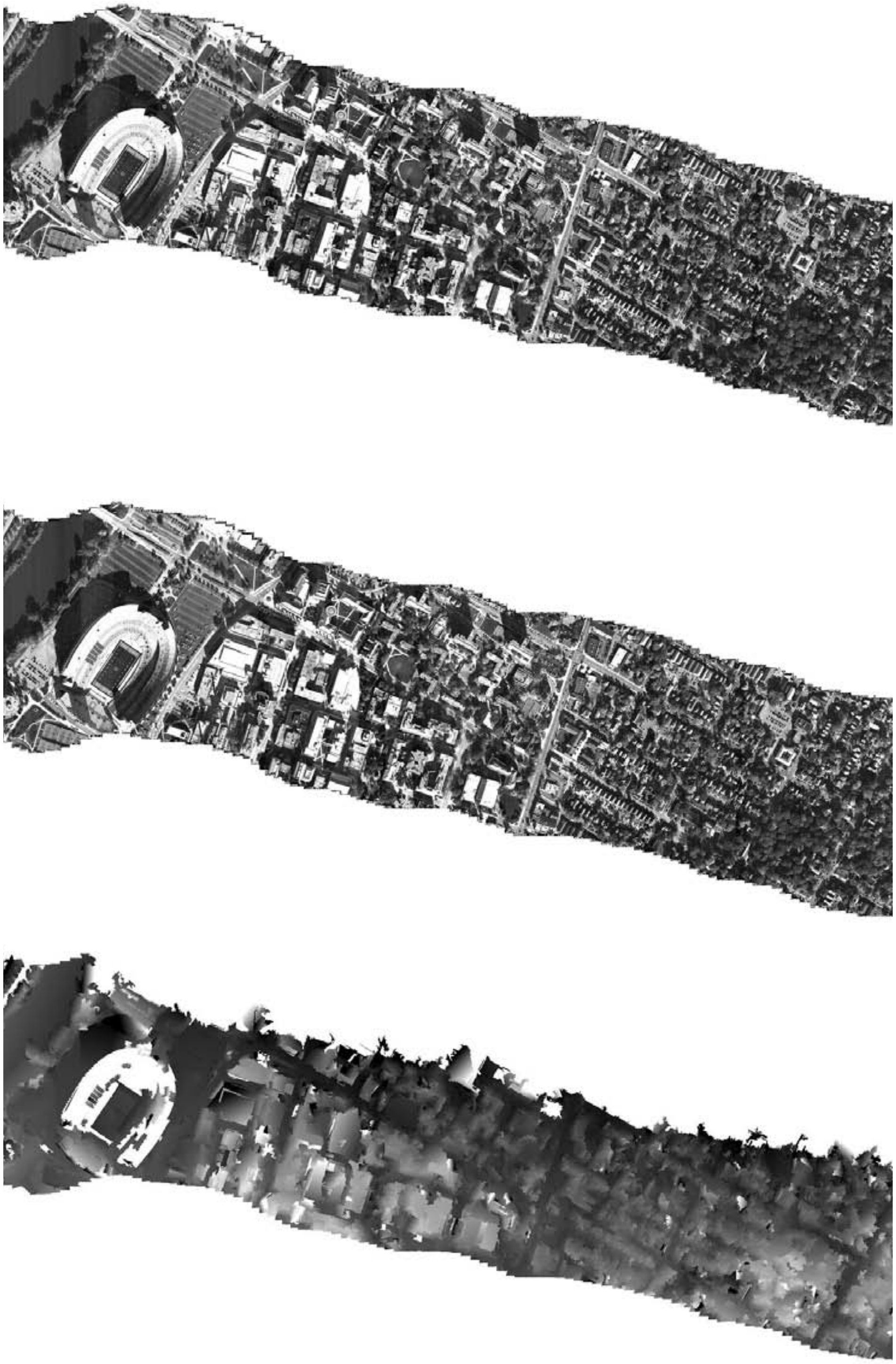

Fig.17. Stereo mosaics and their depth map generated from 246 images (2 frames per second) of the CLIF 2006 dataset. In generating the mosaics, the original CLIF images were down-sampled by 4 times to 1001x668, and the final mosaics are of the size of 4800x1300. The mosaics are again scaled for display.

7.6. Computation time analysis

The two-phase CB3M construction is also efficient in computation time. The following statistics was obtained when our program was run on a PC with Windows XP, an Intel Core 2 Duo 2.0GHz CPU, 4M cache, 3GB memory, 800MHz FSB (BUS). Most of the computation time in the first phase (stereo mosaicing) was spent on orientation estimation using a pyramid-based image registration method, and stereo mosaicing based on the PRISM algorithm. For a typical video sequence with a resolution of 640*480, the speed of the first phase was about 5 Hz (5 frames per second). More analysis on time complexity of image registration can be found in a related paper of ours (Zhu, et al, 2005).

Since this paper is mainly focused on the second phase, we will provide more information for this phase. In this phase, most of the computation time is spent on two steps: segmentation (a pre-processing step to segment the reference image) and matching (the followed step of matching multi-view pushbroom mosaics). The segmentation step was implemented using the mean shift algorithm by Comanicu & Meer (2002) and a toolbox provided by the authors, and the matching step was implemented by us in C++.

Mean shift algorithm is one of the most popular nonlinear clustering algorithms. It has been used to segment an image by clustering the image into color patches and each patch is a cluster in color space (Comanicu & Meer , 2002). The source code is available to download from their website at http://www.caip.rutgers.edu/riul/research/code/EDISON. The toolbox is implemented using C++ with wxWidgets, the latter is an open source, cross-platform GUI and tools library for GTK, MS Windows, and MacOS. Given an image together with a few segmentation parameters (for clustering in color space), the segmentation program will produce a labeled image (display on screen and/or write segmented image onto disk).

Table 3 lists the time performance for the three video sequences we have presented in this paper: the campus scene, the indoor scene, and the NYC scene. For each sequence, the effective size of each mosaic (denoted as M), the number of patches produced in the reference mosaic after segmentation (denoted as N), the search ranges in both the direction of the camera motion, and the perpendicular direction (denoted as S_h and S_v), the number of pairs of pushbroom mosaics (denoted as K) used in each case, and the times spent in both segmentation and matching are listed in the table. Note that in the table, the sizes of the mosaics are the effective sizes that count the real scene pixels, excluding those pixels that are blank in the borders (this is particularly obvious for the campus scene since the mosaics run in a diagonal direction). Apparently, among the two steps (segmentation and matching), much longer time is spent on multi-view stereo matching, which includes the correlation step in local match (Section 5.1), and image warping in match evaluation (i.e., SSD) in the multi-view refinement (Section 5.2) and plane updating using global and local constraints (Section 5.3). Since both local match and image warping are based on patches over multiple mosaics, the match time is therefore a function of the number of patches N, number of pairs of mosaics K, and complexity of the scene (leading to various numbers of interests points). Roughly, the time complexity for patch-based multi-view local match and warping can be estimated as

$$T = O(NKS_hS_v) +O(SK) \qquad (25)$$

where the first term is for local match, which is proportional to the number of patches, the number of mosaic pairs and the search area, while the second term is for the image warping, which is proportional to the effective size of the mosaic (since all the pixels need to be warped to estimate the goodness of stereo match), and the number of mosaic pairs.

The last two columns of Table 3 are the real time spent in segmentation and matching (in seconds), respectively, and the average time (in ms) spent per patch for segmentation, and per patch per pair of mosaics for stereo match. In particular, the average times in match per patch in the three examples are comparable, which are roughly speaking only functions of the corresponding search ranges. Note that we have not optimized the code for computational efficiency for correlation and warping, which could be implemented using look-up-table and integer iteration techniques that will greatly improve the time performance.

Table 3. Computation time analysis

Clips	Effective Size of mosaic (M)	# of patches (N)	# of mosaic pairs (K)	Search Range (S_h,S_v)	Segmentation time (Ts in seconds, and Ts/N in ms)		Matching time (Tm in seconds, and Tm/(NK) in ms)	
					Ts	Ts/N	Tm	Tm/(NK)
Campus	3900x700	15298	8	(8, 7)	44	2.88	5973	48.81
Indoor	3850x250	3204	10	(15, 3)	3	0.94	745	23.25
NYC	3700x2000	37166	3	(30, 8)	330	8.88	9420	84.49

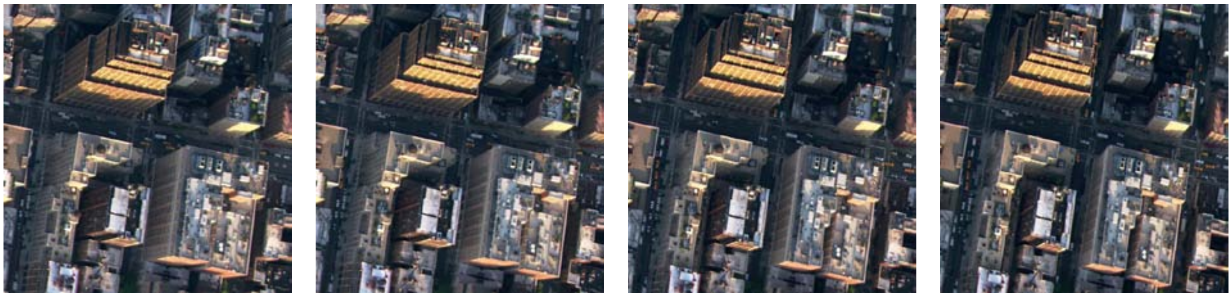

Fig. 18. Four parallel-perspective views with different viewing angles over a close-up of the New York City scene. Parallax is preserved as well as the dynamic motion of moving vehicles.

8. Image-Based Rendering Using Stereo and 3D Mosaics

We have shown a system that can take video sequences of large-scale scenes as input and is able to produce large field-of-view pushbroom mosaics, as well as reconstructed 3D data and dynamic object detection. In the following sections we describe how the multi-view mosaics and CB3M data are used and visualized now and how these may be delivered over the Internet.

8.1. Mosaic-based rendering

The first output we create after phase I is a set of multi-view pushbroom mosaics, as illustrated in Figure 1. We have applied our system to a minute long (HD – 1080p) aerial video of New York City. Fig. 18 shows four close-up views of the (8 total) mosaics that were generated for the scene taken over New York City. The first mosaic close-up is created by one of the leading scanline's (forward looking relative to the motion) and the fourth mosaic close-up is created by a trailing scanline (backward looking). It can be visually inspected that the motion parallax of structures are preserved and the scene is aligned (note that these four mosaic close-ups represent novel parallel-perspective views from viewing angles that are far apart). There are two ways in which we can view the mosaics:

Mosaic viewing Viewing the mosaics independently already provides an efficient representation and summary of the scene being imaged. By stacking all of the mosaics generated and flipping among them we can observe that object movements and parallax are also preserved. We have thus efficiently represented a compressed 150MB, 1 minute long video in a set of 8 mosaics encoded in JPEG, with an average size of 2.5MB each. (Molina, et, al, 2008 – See nyc-2d.mov)

Stereo viewing The multi-view mosaics can also be rendered to view the 3D data using cyan-red glasses (Molina, et, al, 2008 – See nyc-anaglyph.mov) or shutter glasses. This is possible since the mosaics generated are aligned and in stereo correspondence. We have created a desktop application (Fig. 19) that loads all of the mosaics and combines two views at a time to create an anaglyph by using the red channel from one mosaic and the green and blue channels of another. The application allows us to set the 'disparity' defined as the distance between the two consecutive mosaic views used, and the viewing angle by allowing the user to gradually change which two mosaics are being viewed (using viewing 'position' slider in Fig. 19). In addition we can pan and zoom the mosaic. The application also works with shutter glasses and can be potentially used with polarized glasses.

Recently 3D-DLP display technology has become available in consumer rear-projection and LCD televisions. The technology is now inexpensive and will soon become a standard feature on televisions from major manufacturers. Standardization committees are working on formats for Video, HDMI and Blu-Ray Discs to support 3D content. Our visualization application is already working on these displays and demonstrates an improvement in 3D perception. The major benefit with 3D-DLP displays is that they run at a 120 Hz refresh rate (more expensive consumer models are already shipping at 240 Hz). The shutter glasses then alternately turn on and off, reducing the rate to 60 Hz for the viewer which is high enough to prevent the perception problems the last generation of the technology had. In addition the shutter glasses allow the viewers to see the full color of the 3D content; with cyan-red glasses there is a major loss of color due to the color filtering performed by the glasses.

Our pushbroom mosaicing algorithm and mosaics do not require any changes to work on new 3D displays. Only the method of displaying the images is modified. Stereo viewing of multi-view mosaics works well and can be further improved by applying the techniques presented and evaluated by Ideses and Yaroslavsky (2005).

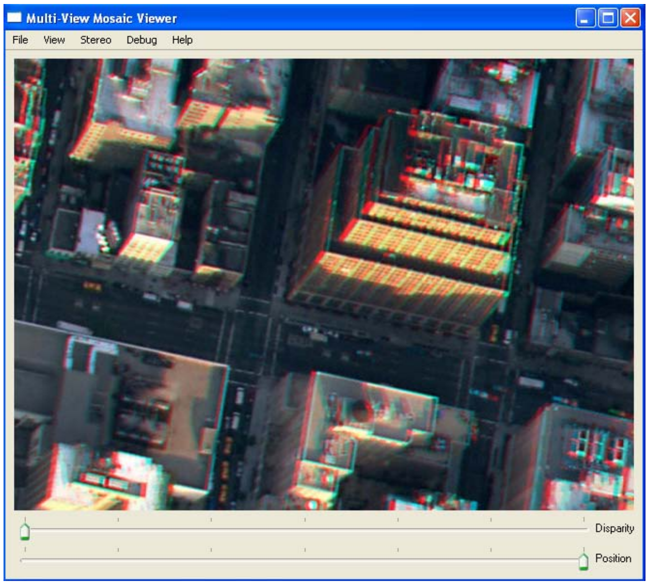

Figure 19. Screenshot of Multi-View Mosaic Viewer displaying a cyan-red anaglyph.

8.2. Content-based mosaic rendering

True 3D models can be visualized by rendering the 3D plane patches described in the CB3M reconstruction of scenes. In addition, it is possible to texture map the 3D models by using the multi-view pushbroom mosaics. On the desktop, scenes can be rendered using OpenGL or DirectX. A rendered 3D model will allow for additional information to be overlaid, such as images and videos, and hot spots that link to related websites or media.

Rendering in true 3D over the web enables applications like those described above to be built. But while 3D formats for web delivery exist, 3D content is lacking, and web browser plug-ins and 3D browsers do not yet have a large user base. The CB3M format we use is an efficient format for storing and sharing large-scale 3D scenes, but is not meant to provide all of the additional features that standard formats provide. Although, it is possible to convert our CB3M data to other formats as necessary.

Various 3D formats exist for the Internet, many are proprietary or have evolved from products and others are open standards. Many websites and companies have used the formats but they still have not reached the market penetration other rich content formats on the web (such as Flash) have reached. VRML and its successor X3D are both ISO standards and development is being led by the Web3D Consortium. VRML (Virtual Reality Markup Language) was the first standard for 3D content on the Web and it has been widely used among 3D applications. X3D (which is based on XML) is the more recent ISO standard and successor to VRML. Many 3D applications can export to the VRML and X3D formats. Both VRML and X3D allow for scripting and animation of 3D models.

COLLADA is another XML based open format that is being developed as an industry standard for 3D data exchange. COLLADA's development is being led by the Khronos Group which also controls many other open standards, which include OpenGL. The COLLADA specification was not designed for the web, but it can be used in conjunction with X3D and provides many features for representing 3D data. Google Earth allows importing and exporting the COLLADA format in addition to its own KML format.

9. Generating Mosaics with Circular Camera Path

In the following, we will discuss the generation of stereo mosaics under circular camera path. Eipipolar geometry of circular pushbroom stereo mosaics and issues on mosaic generation will be discussed. By combining the models under both linear motion and circular motion, we hope to pave the way to establish stereo mosaic models under more general motion in the future.

9.1. Ideal imaging geometry

A 1D camera with a single column off the center of an angle β, moves along a circular path with a center C and a radius R. The camera's optical axis Z is perpendicular to the circular path. A circular "panoramic" image is generated by this scanning camera (Fig. 20). The scanlines of the circular images are circles. Such an image is represented by I(α,r), where α is the angle of the pixel along the circular measured from a starting point, and r is the distance along the column direction.

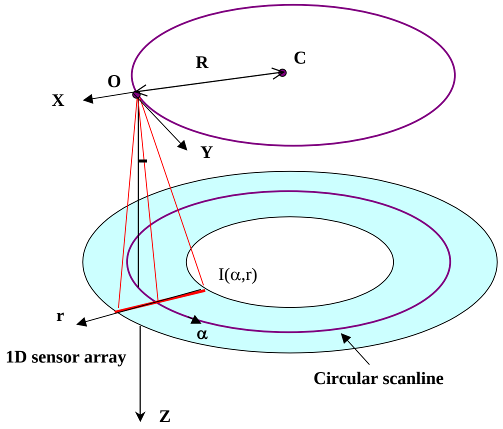

Fig. 20. Circular imaging geometry

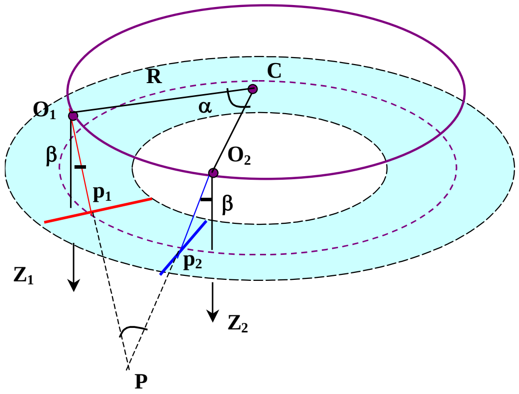

Figure 21. Circular stereo geometry

9.2. Stereo correspondences and 3D reconstruction

Given two such circular scanning cameras moving on the same circular path with a center C and a radius R, both with viewing angle β, one looking forward (O1) and the other looking backward (O2), a pair of circular stereo panoramas can be generated (Fig. 21). For any 3D point P, its correspondences, $p_1(\alpha_1,r_1)$ and p_2 (α_2,r_2), in the two panoramas are (approximately) along a circular scanline. Therefore, we have r1 = r2, and the angular disparity

$$\alpha=\alpha_2 - \alpha_1. \tag{26}$$

The baseline B between the two views O_1 and O_2 can be calculated as

$$B = 2R \sin(\alpha/2) = R\alpha, \tag{27}$$

where α is measured in radian, and the radius is much larger than the arc length B. Then the distance from each view (O1 or O2) to the 3D point can be calculated as

$$D = R \sin(\alpha/2) / \sin\beta = R\alpha /(2\beta)$$

where is also measured in radian. Hence the Z coordinate of the point P can be computed as

$$Z = D \cos\beta = R \sin(\alpha/2)/\tan\beta = 2R\alpha /\tan\beta \qquad (28)$$

We want to note here that pushbroom stereo mosaics under circular motion path is different from multi-perspective stereo panoramas with circular projections (Peleg, et al, 2001; Shum & Szeliski, 1999). In stereo panoramas with circular projections, the optical axis of the camera points to the center of the circular motion, while in pushbroom stereo mosaics, the optical axis of the camera is perpendicular to the circular motion path. In fact, in all the cases where the optical axis is not pointing to the center of the circular path, pushbroom stereo mosaics can be generated by applying image rectification before mosaicing.

Second, in circular pushbroom stereo mosaics, depth error is independent of the depth in theory (Eq. 28), which is the same as linear pushbroom stereo mosaics. Therefore, the two types of pushbroom stereo mosaics can be combined into one model for a more general motion, in that the motion is characterized by piecewise linear and circular. Then if a camera moves on a more general path, a generalized pushbroom mosaic can be built along that path, in which the projection is perspective perpendicular to the direction of the motion. If the rates of changes of motion directions are slow, we can fit the path with smooth, piecewise linear and circular (with large radii), so that locally the epipolar geometry is still along “scanlines”.

9.3. Stereo mosaicing from video with circular camera path

In the following we discuss how we can generate such circular mosaics from an aerial video sequence when a camera undergoes a circular path. In theory, a pair of stereo mosaics can be created by extracting two columns, with the same viewing angle β, one looking forward and the other looking backward. However, there are several practical issues. We assume the camera moves on a perfect circular path, or the circle is large enough to compensate the small drifts of the camera from the viewing circle by 2D image translations. The camera may not have a nadir viewing angle, so an orthorectification step is carried out before mosaicing. Finally, the camera views may not be dense enough to generate a dense and seamless circular mosaic if only one column is extracted.

We will describe the steps to create one such mosaic, with a viewing angle β. A pair of stereo mosaics can be generated in the same way.

Step 1. Camera orientation estimation.
This can be done by using either INS/GPS or bundle adjustment techniques. We realize this is an important step, but in this project, we will mainly focus on the geometric models and the stereo mosaicing algorithms. There is a large body of literature discussing this issue.

Step 2. Camera path generation and image rectification.
By fitting the viewpoints of the moving camera, a circular path (with center C and radius R) can be found. Then the images in the sequence is transformed (by rotation and translations) so that each rectified image has a viewpoint on

the circular path, and has an optical axis perpendicular to the circular path. Then the rectified sequence is ready for use in creating mosaics.

Step 3. Circular image transformation

By using the information of the circular path, each image should be transformed from its rectilinear representation to a circular representation (Fig. 22). This step can be combined with Step 2.

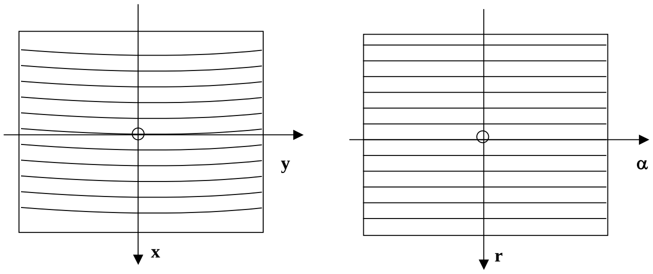

Fig. 22. From rectilinear to circular images

Step 4. Circular panorama generation.

Given two views, O1 and O2, two rays (with circular angles α_1 and α_2 determined by the two view locations) can be directly cut from column b in each image, and paste to the mosaics. The circular angle between them is α. Between these two columns, finding the correspondences β_1 and β_2 of a 3D point P, in the two images respectively, the distance of the point P can be calculated as

$$Z = R\,\alpha / (\beta_1 - \beta_2) \quad (29)$$

Then the circular angle α_i of the reprojected ray to the viewing direction of β is

$$\alpha_i \;= Z\,(\beta_1 - \beta) \;= R\,\alpha\,(\beta_1 - \beta) / (\beta_1 - \beta_2) \quad (30)$$

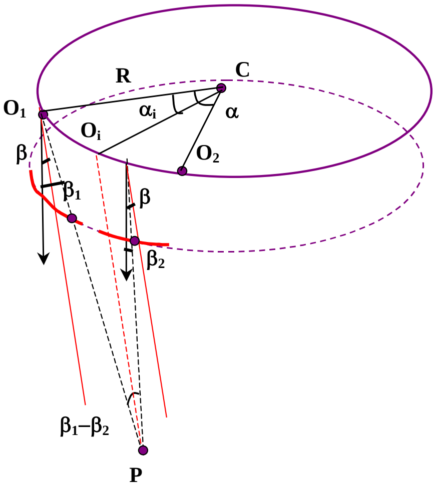

Fig. 23. Circular mosaic generation

Note that Eq. (30) indicated that this procedure is strikingly similar to the parallel ray interpolation for stereo mosaicing (PRISM) we have proposed (Zhu, et al, 2004), therefore the same PRISM algorithm with slight modification can be directly applied for both linear pushbroom mosaicing and circular pushbroom mosaic, and then to the generalized pushbroom mosaicing. A dense 2D mosaic can be generated by image area triangulation and interpolation (Zhu, et al, 2004). Furthermore, linear pushbroom mosaics are a special case of circular pushbroom mosaics when the radius of the circular path is infinite.

9.4. Experiments and preliminary results

Three data sets have been used to evaluate our circular mosaic algorithm. The first set is a simple simulation with perfect circular camera motion. The camera has a nadir view of the scene at an altitude of 600 meters moving in a circle with radius 156.4 meters. The camera's FOV is 15.189 degrees. The second data set was a simulation provided by Numerica. In this simulation the camera traveled at an altitude of 2000 meters in a circular path with a 2000 meter radius. The camera's FOV is 2 degrees and was fixated on the center of the scene. This simulation did not have perfect consistent angles and motion on the scene. The third set of data was taken from the CLIF 2007 dataset. Sequences of 360 degree circular paths were extracted from camera 0; in particular they are frames '000000-100075' to '000000-100305', a total of 231 frames.

Simple simulation

In this simulation the center of the camera path is known and the same for all frames. We go directly to Step 3 (circular image transformation) in our algorithm. Fig. 24 shows the original camera's image and it transformed into a circular projection image using a Cartesian to Polar coordinate transformation where the center is at (320,-386) in pixels. (Note: -386 in the Y-axis refers to center being 386 pixels above the image.)

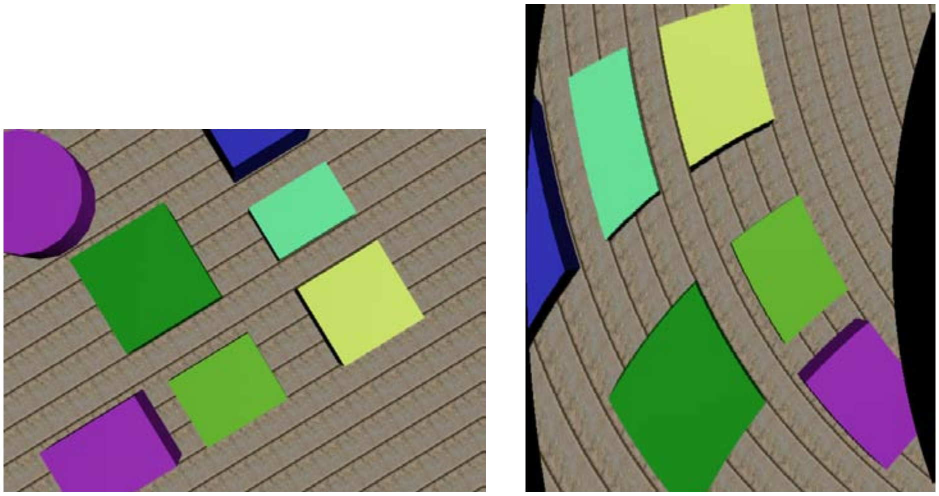

Fig. 24 On the left is the original frame in Cartesian X,Y coordinates;
On the right is the transformed image in Polar R,θ coordinates.

In the original sequence the camera motion was circular and counter-clockwise in Cartesian coordinates. After Step 3 the sequences radii of the images are registered and the motion is only in the angle θ; hence in the transformed sequence, this is a linear motion. We can do Step 4 (circular panorama generation) on the transformed sequence using our pushbroom mosaicing algorithm. In addition this allows us to generate multiview pushbroom mosaics. Fig. 25 shows a "left" view and a "right" views of the simulated scene.

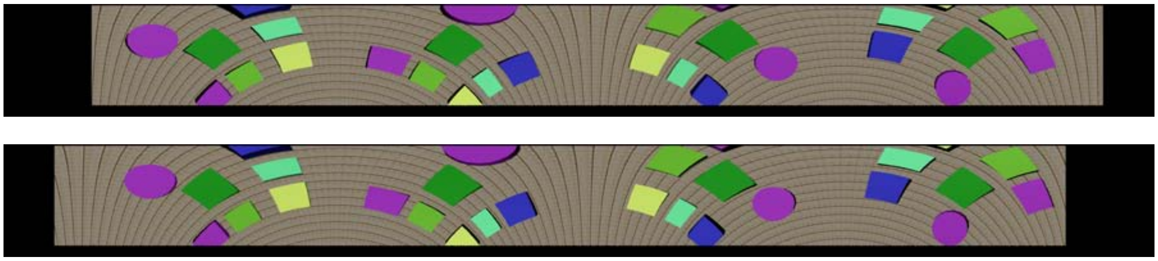

Fig. 25. The top/bottom views show left/right mosaics of the scene. The Y-axis represents radius, and the X-axis represents degrees. Both mosaics show 360 degrees of coverage.

Fig. 26 shows one of the left view mosaics transformed back to Cartesian coordinates, which shows an undistorted global view of the original scene.

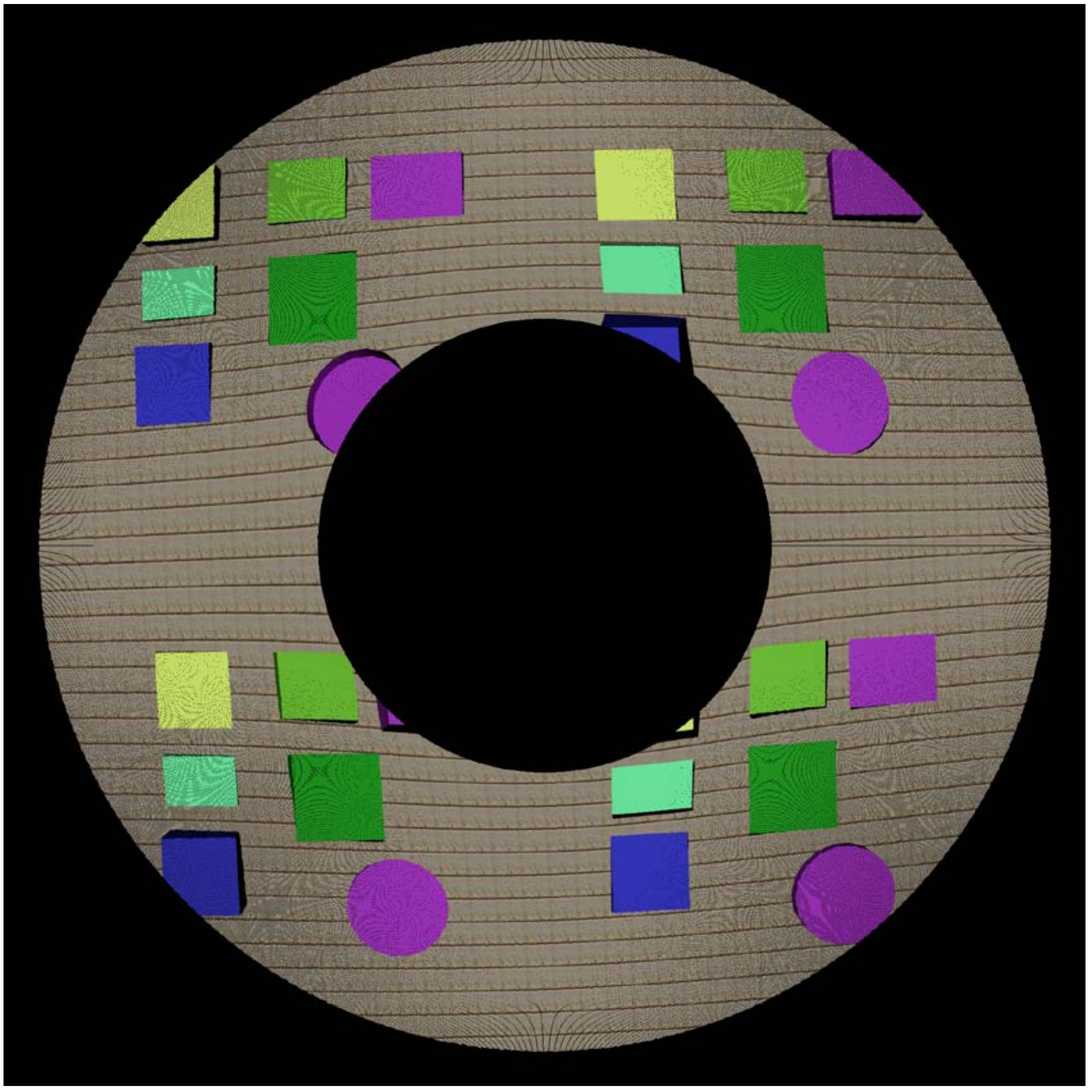

Fig. 26. One of the left view mosaics transformed back to Cartesian coordinates

Numerica simulation results

The Numerica simulation was not a perfect circular camera motion case. For this sequence a very fast and coarse registration was applied to estimate the center of the camera's path. Then again the images are warped into the polar coordinate system so that the motion of the polar sequence is a linear motion and our pushbroom mosaicing program can be applied to the warped sequence. Fig. 27 shows one of the circular mosaics and its conversion back to Cartesian coordinates.

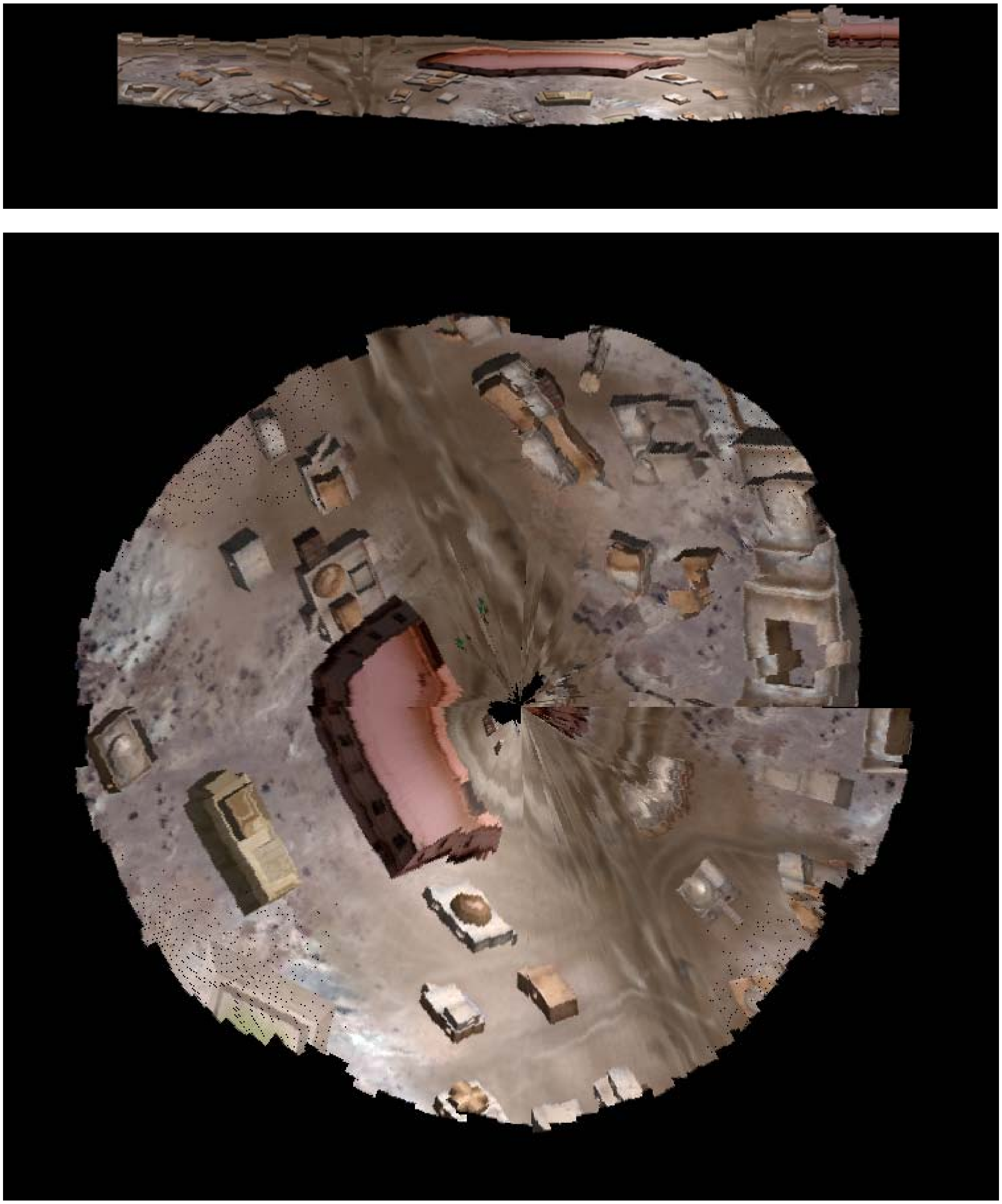

Fig. 27. A polar mosaic (top) of the Numerica simulation and its Cartesian conversion (bottom).

Note that in the Cartesian conversion image, the head and tail are not aligned very well due to the registration error. These results were all derived from image based techniques. The result can be further improved by applying image rectification from image based techniques (not done in this test) or using INS/GPS data which is available in this sequence.

CLIF 2007 results

These results are for a circular camera path sequence in camera 0 of the CLIF 2007 dataset. These results were attained without any image rectification prior to Step 3. Only an image based registration is applied for Step 4 on the transformed imagery. An estimate of the camera path center was used for the following results.

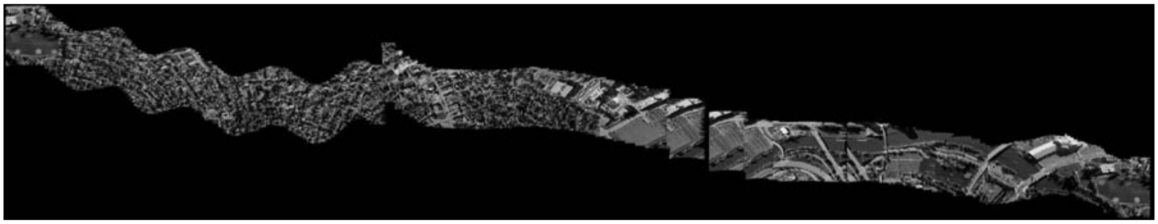

Fig. 28. A polar representation of a mosaic of full 360 degrees.

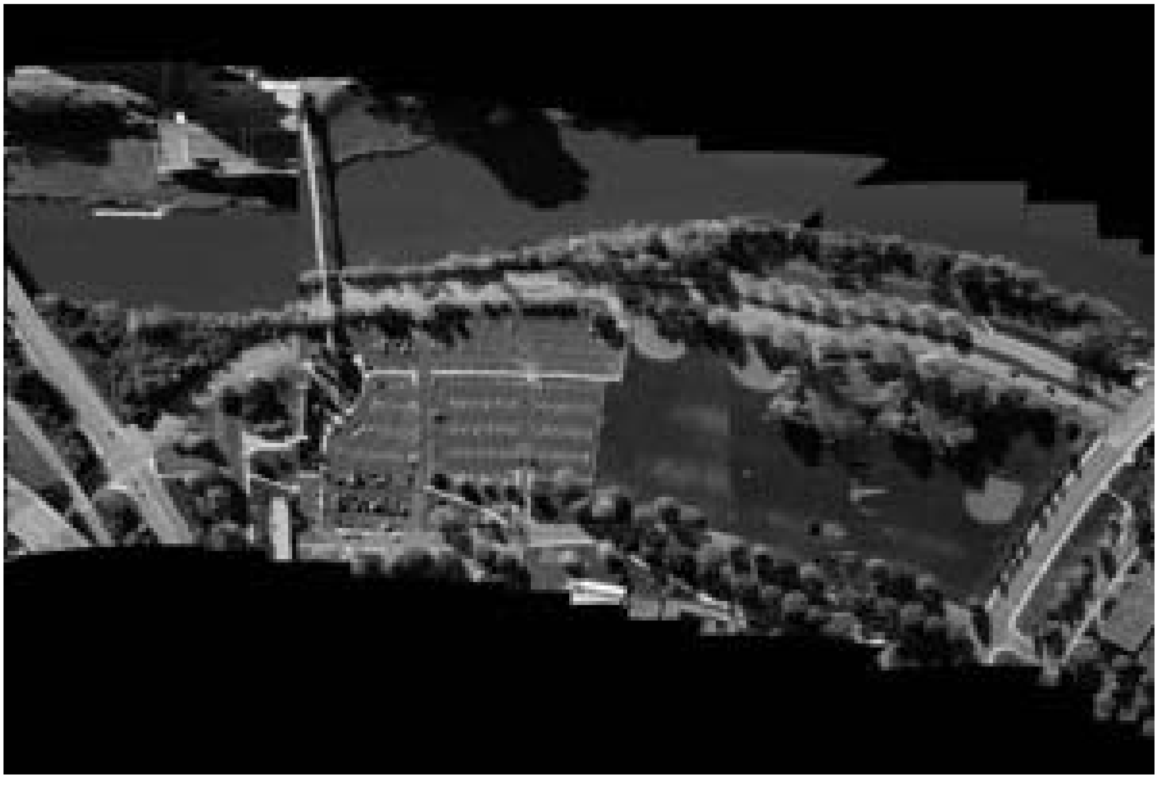

Fig. 29. A cropped portion of the right end of the scene above.

In this sequence we noted that the camera moved significantly more between frames in the middle of the sequence. While a registration was still attained, it did not work as well for pushbroom mosaicing (this is why we see gaps in the full mosaic). Using a higher frame rate would improve this problem.

10. Concluding Remarks

In this work we propose to construct a content-based 3D mosaic representation (CB3M) for long video sequences of 3D and dynamic scenes captured by a camera on a mobile platform. In real applications, the motion of the camera should have a dominant direction of motion (as on an airplane or ground vehicle), but 6 DOF motion is allowed. A two-phase approach is presented to create a CB3M representation. In the first phase, multiple parallel-perspective (pushbroom) mosaics are generated to capture both the 3D and dynamic aspects of the scene under the camera coverage. In the second phase, a multi-view, segmentation-based stereo matching approach is applied to extract parametric representation of the color, structure and motion of the dynamic and/or 3D objects, and to represent them as planar surface patches.

The content-based 3D mosaic (CB3M) representation is a highly compressed visual representation for very long video sequences of dynamic 3D scenes. It could fit into the MPEG-4 standard, in which a scene is described as a composition of several Video Objects (VOs), encoded separately. The compression of a video sequence comes from both steps: stereo mosaicing and then content extraction. For both simulated and real image sequences of large-scale cultural scenes with many man-made buildings and vegetations, with more than 1000 frames of 640*480 color images, a compression ratio of thousands to ten thousands is achieved. More importantly, the CB3M representation has object contents represented, which provides the following benefits for many applications, such as urban transportation planning, aerial surveillance and urban modeling. The *panoramic mosaics* provide a synopsis of the scene with all the 3D objects and dynamic objects in a single view. The *3D contents* of the CB3M representation make further object recognition and higher-level feature extraction possible. The *motion contents* of the CB3M representation provide dynamic measurements of moving targets in the large-scale scene.

We will continue to work on two directions in advancing and extending the technologies proposed in this report. First, in the CB3M representation presented in this paper, however, many details and practical issues have not been considered. First, more experiments are needed with both simulated and real video sequences to evaluate the coding and compression capabilities of this representation. Second, in order to use the CB3M representations for real applications, further enhancements are also needed. For example, in the current implementation, only 3D parametric information of planar patches in a single reference mosaic is obtained. Since different visibilities are shown in mosaics with different viewing directions, we want to extend the approach presented in the paper to produce multiple depth maps with multiple reference mosaics and then integrate the results by performing occlusion analysis. Third, developing higher-level representations that group the lower-level natural patches for physical objects may also be very useful for many applications (Li & Zhu, 2008). For example, the neighboring regions, which have been extracted in the patch and interest point extraction stage, and which are important in object recognition and occlusion handling in image rendering, are not represented in the current model. Finally, in our experiments, we only handled those moving objects that do not obey the epipolar geometry of the pushbroom stereo, i.e. they do not move in the direction of the camera. Otherwise we have to assume they move on a ground plane. This is mostly valid for aerial videos, but for ground video sequences captured on a ground vehicle for scenes with other moving vehicles and humans, the method proposed at the end of Section 4 should be applied. This also requires further analysis of relations of object regions (patches). Even with aerial videos, it will be useful to be able to detect moving targets in air and other erratic behaviors, for example people walking on top of parking garages.

Second, we would like to generalize the pushbroom stereo mosaicing approach with more general camera motion. For example, as we have briefly discussed in Section 9, we are working on stereo mosaicing with circular camera motion, and we have derived a geometric model for such a case. However, the experimental results are still very preliminary. More implementation efforts are needed for algorithmic details. In the long term, we would also like to combine pushbroom stereo mosaicing techniques in linear and circular motion cases, and generalize them to situations with more general camera motion paths. We also realize that camera orientation estimation with many video frames is still a challenging issue, and we hope that the results of this paper will stimulate more interests in the research and development of this problem.

11.References

[1]. Boykov, Y., Veksler, O. and Zabih, R. 2001. Fast approximate energy minimization via graph cuts, *IEEE Trans. Patten Analysis and Machine Intelligence*, Vol. 23, No. 11.

[2]. Chai, J. and Shum, H –Y. 2000. Parallel projections for stereo reconstruction. In *Proc. Computer Vision and Pattern Recognition (CVPR'00)*: II 493-500.

[3]. Comanicu, D. and P. Meer, 2002. Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Patten Analysis and Machine Intelligence*, May 2002

[4]. Cornelis, N., Leibe, B., Cornelis, K. and Van Gool, L. 2008. 3D urban scene modeling integrating recognition and reconstruction, *Int. J. Computer Vision*, 78 (2-3), July: 121-141

[5]. Coughlan, C. and Yuille, A. 1999. Manhattan world: compass direction from a single image by Bayesian inference. In *Proc. International Conference on Computer Vision (ICCV'99)*, 941-947.

[6]. Deng, Y., Yang, Q., Lin, X. and Tang, X. 2005. A symmetric patch-based correspondence model for occlusion handling. In *Proc. International Conference on Computer Vision (ICCV'05)*, II: 1316-1322.

[7]. Dickson, P., Li, J., Zhu, Z., Hanson, A. R., Riseman, E. M., Sabrin, H., Schultz, H.and Whitten, G. 2002. Mosaic generation for under-vehicle inspection. In *Proc. IEEE Workshop on Applications of Computer Vision*, Dec 3-4

[8]. Fusiello, A., Roberto, V. and Trucco, E. 1997. Efficient stereo with multiple windowing. In *Proc. Computer Vision and Pattern Recognition (CVPR'97)*: 858-863

[9]. Gupta R and Hartley R, 1997. Linear pushbroom cameras. *IEEE Trans. Pattern Recognition and Machine Intelligence*, 19(9): 963-975

[10]. Herman, T. and Kanade, T. 1984. The 3D MOSAIC scene understanding system: incremental reconstruction of 3D scenes from complex images. *Tech. Report*, Robotics Institute, Carnegie Mellon University.

[11]. Herman, M. and T. Kanade,T.1986. Incremental reconstruction of 3d scenes from multiple complex images. *Artificial Intelligence*, Vol. 30, pp. 289-341.

[12]. Hsu, S. and Anandan, P., 1996. Hierarchical representations for mosaic based video compression, In *Proc. Picture Coding Symp.*, 395-400

[13]. Ideses, I., and L. P. Yaroslavsky, 2005. "Three Methods that improve the visual quality of colour anaglyphs", J. Opt. A: Pure Appl. Opt. 7 755-762.

[14]. Irani, M., Anandan, P., Bergen, J., Kumar, R. and Hsu, S., 1996. Mosaic representations of video sequences and their applications. *Signal Processing: Image Communication*, vol. 8, no. 4, May.

[15]. Kanade, T. and Okutomi, M. 1991. A stereo matching algorithm with an adaptive window: theory and experiment, In *Proc. IEEE International Conference on Robotics and Automation (ICRA'91)*, II: 1088-1095

[16]. Ke, Q. and Kanade, T. 2001. A subspace approach to layer extraction, In *Proc. Computer Vision and Pattern Recognition (CVPR'01).*

[17]. Koenen, R.., Pereira, F. and Chiariglione, L. 1997. MPEG-4: Context and objectives. *Signal Processing: Image Communications*, 9(4):295-300.

[18]. Kolmogorov, V. and Zabih, R., 2001. Computing visual correspondence with occlusions using graph cuts, In *Proc. International Conference on Computer Vision (ICCV'01)*, Vol. I:508-515

[19]. Koschan, A., Page, D., Ng, J.-C., Abidi, M., Gorsich, D. and Gerhart, G., 2004. SAFER under vehicle inspection through video mosaic building, *Int. J. Industrial Robot*, September, 31(5): 435-442

[20]. Leung, W. H. and Chen, T. 2000. Compression with mosaic prediction for image-based rendering applications, In *Proc. IEEE Int.. Conf. Multimedia & Expo.*, New York, July.

[21]. Li, Y., Shum, H.-Y., Tang, C.-K. and Szeliski, R. 2004. Stereo reconstruction from multiperspective panoramas. *IEEE Trans Pattern Analysis and Machine Intelligence*, 26(1): pp 45-62.

[22]. Li, X. and Zhu, Z., 2008. Automatic Object Classification through Semantic Analysis, *the 20th IEEE International Conference on Tools with Artificial Intelligence*, 3-5 Nov. 2008, Vol 2: 497-504

[23]. Medioni, G. and Kang, S. 2004. *Emerging Topics in Computer Vision*. Prentice Hall, ISBN: 0131013661.

[24]. Molina, E., H. Tang, Z. Zhu, O. Mendoza, 2008. Mosaic-based Modeling and Rendering of Large-Scale Dynamic Scenes for Internet Applications, *NAECON 2008 - National Aerospace and Electronics Conference*, Dayton, Ohio, USA, Jul 16-18. http://visionlab.engr.ccny.cuny.edu/~molina/publications/08/nyc-videos.zip

[25]. Noble, A., Hartley, R. Mundy, J. and Farley, J. 1994. X-Ray Metrology for Quality Assurance, In *Proc. IEEE Int. Conf Robotics and Automation (ICRA'94)*, II pp 1113-1119

[26]. Odone, F., Fusiello, A. and Trucco, E. 2000. Robust motion segmentation for content-based video coding, In *Proc. 6th Conference on Content-based Multimedia Information Access*, College de France: 594-601.

[27]. Okutomi M. and T. Kanade, 1993. A multiple-baseline stereo, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 4, pp. 353-363.

[28]. Peleg, S., Ben-Ezra, M. and Pritch, Y., 2001. Omnistereo: panoramic stereo imaging, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(3): 279-290

[29]. Pollefeys, M., et al. 2008. Detailed real-time urban 3D reconstruction from video, *Int. J. Computer Vision*, 78 (2-3), July: 143-167

[30]. Rav-Acha, A., Engel, G. and Peleg, S. 2008. Minimal aspect distortion (MAD) mosaicing of long scenes. *Int. J. Computer Vision*, 78 (2-3), July : 187-206

[31]. Scharstein, D. and Szeliski, R. 2002. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *Int. J. Computer Vision*, 47(1/2/3): 7-42, April-June .

[32]. Shum, H.-Y. and Szeliski, R. 1999. Stereo reconstruction from multiperspective panoramas. In *Proc. International Conference on Computer Vision (ICCV'99)*: 14-21

[33]. Sun, C. and Peleg, S. 2004. Fast Panoramic Stereo Matching using Cylindrical Maximum Surfaces, *IEEE Trans. System, Man and Cybernetics*, Part B, 34, Feb.: 760-765.

[34]. Sun, J. Zheng, N. and Shum, H. 2003. Stereo Matching Using Belief Propagation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(7), July

[35]. Tao, H., Sawhney, H. S. and Kumar, R. 2001. A global matching framework for stereo computation. In *Proc. International Conference on Computer Vision (ICCV'01), I 532-539*

[36]. Tang, H., Z. Zhu, G. Wolberg and J. R. Layne, 2006. Dynamic 3D Urban Scene Modeling Using Multiple Pushbroom Mosaics, the Third International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT 2006), University of North Carolina, Chapel Hill, USA, June 14-16.

[37]. Tang H., and Z. Zhu, 2008. Exploiting Local and Global Scene Constraints in Modeling Large-Scale Dynamic 3D Scenes from Aerial Video, Workshop on Search in 3D (S3D), June 27, 2008. In conjunction with IEEE Conference on Computer Vision and Pattern Recognition (CVPR2008).

[38]. Triggs, B., McLauchlan, P., Hartley, R. and Fitzgibbon, A. 2000. Bundle Adjustment - A Modern Synthesis, In *Vision Algorithms: Theory and Practice*, Lecture Notes in Computer Science, vol 1883, pp 298-372, eds. B. Triggs, A. Zisserman and R. Szeliski", Springer-Verlag.

[39]. Xiao, J. and Shah, M. 2004. Motion layer extraction in the presence of occlusion using graph cut. In *Proc. Computer Vision and Pattern Recognition (CVPR'04)*

[40]. Zheng, J. Y. and Tsuji, S. 1992. Panoramic Representation for route recognition by a mobile robot. *Int. J. Computer Vision*, 9(1), pp. 55-76

[41]. Zheng, J.Y., and Shi, M. 2008. Scanning depth of route panorama based on stationary blur. *Int. J. Computer Vision*, 78 (2-3), July :169-186

[42]. Zhou, Y. and Tao, H. 2003. A background layer model for object tracking through occlusion. In *Proc. International Conference on Computer Vision (ICCV'03)*: 1079-1085.

[43]. Zhu, Z., Hanson, A. R., Schultz H. and Riseman, E. M., 2003. Generation and error characteristics of parallel-perspective stereo mosaics from real video, book chapter in *Video Registration*, M. Shah and R. Kumar (Eds.), Video Computing Series, Kluwer Academic Publisher, Boston, May: 72-105

[44]. Zhu, Z. and Hanson, A. R. 2004. LAMP: 3D layered, adaptive-resolution and multi-perspective panorama - a new scene representation. *Computer Vision and Image Understanding*, 96(3), Dec: 294-326.

[45]. Zhu, Z., Riseman, E. M. And Hanson, A. R. 2004. Generalized Parallel-Perspective Stereo Mosaics from Airborne Videos, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(2), Feb: 226-237

[46]. Zhu, Z., Riseman, E. M., Hanson, A. R. .and Schultz, H., 2005. An efficient method for geo-referenced video mosaicing for environmental monitoring. *Machine Vision and Applications*, 16(4): 203-126

[47]. Zhu, Z. and Hanson, A. R. 2006. Mosaic-based 3d scene representation and rendering. *Signal Processing: Image Communication*, Elsevier, 21(6), Oct: 739-754

[48]. Zhu, Z. and Hu, Y.-C., 2007. Stereo Matching and 3D Visualization for Gamma-Ray Cargo Inspection, In *Proc. IEEE Workshop on Applications of Computer Vision*, Feb 21st-22nd, Austin, Texas, USA

Appendix: Publications Under the Support of This Grant

1. H. Tang and Z. Zhu, **Content-Based 3D Mosaics for Representing Videos of Dynamic Urban Scenes,** *IEEE Transactions on Circuits and Systems for Video Technology*, accepted, August 2008.
2. Z. Zhu, Y.-C. Hu and L. Zhao, **Gamma/X-Ray Linear Pushbroom Stereo for 3D Cargo Inspection**, *Machine Vision and Applications*, October 2008, Online First at http://dx.doi.org/10.1007/s00138-008-0173-8
3. Z. Zhu, **Mobile Sensors for Security and Surveillance**, *Journal of Applied Security Research,* the Haworth Press, vol 4, no 1&2:79–100, January 2009 (invited paper).
4. E. Molina, Z. Zhu, O. Mendoza-Schrock, **Mosaic-based 3D Scene Representation and Rendering of Circular Aerial Video**", *SPIE Defense, Security, and Sensing Symposium, "Evolutionary and Bio-Inspired Computation: Theory and Applications IV" Conference,* April 5-9, 2010 (Accepted)
5. H. Tang, Z. Zhu and J. Xiao, **Stereovision-Based 3D Planar Surface Estimation for Wall-Climbing Robots**, *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 11-15, 2009, St. Louis, USA
6. H. Tang and Z. Zhu, **Exploiting Local and Global Scene Constraints in Modeling Large-Scale Dynamic 3D Scenes from Aerial Video**, *Workshop on Search in 3D (S3D)*, June 27, 2008. In conjunction with *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. http://dx.doi.org/10.1109/CVPRW.2008.4563035
7. X. Li and Z. Zhu, **Automatic Object Classification through Semantic Analysis**, *the 20th IEEE International Conference on Tools with Artificial Intelligence*, 3-5 Nov. 2008, Vol 2: 497-504
8. E. Molina, H. Tang, Z. Zhu, O. Mendoza, **Mosaic-based Modeling and Rendering of Large-Scale Dynamic Scenes for Internet Applications**, *NAECON 2008 - National Aerospace and Electronics Conference*, Dayton, Ohio, United States, Jul 16-18, 2008
9. Z. Zhu and T. Kanade, **Editorial: Modeling and Representations of Large-Scale 3D Scenes**, *International Journal of Computer Vision*, Volume 78, Numbers 2-3 / July, 2008. http://dx.doi.org/ 10.1007/s11263-008-0128-6
10. Z. Zhu, G. Wolberg, J. R. Layne, Dynamic **Pushbroom Stereo Vision for Surveillance and Inspection**, Chapter 8 in *3D Imaging for Safety and Security* , eds. A. Koschan, M. Pollefeys, and M. Abidi, Kluwer/Springer, August 2007, pp 173-200.
11. Z. Zhu, Y.-C. Hu, **Stereo Matching and 3D Visualization for Gamma-Ray Cargo Inspection**, *Proceedings of the Eighth IEEE Workshop on Applications of Computer Vision*, Feb 21st-22nd, 2007, Austin, Texas, USA
12. Zhigang Zhu, Allen Hanson, **Mosaic-based 3D Scene Representation and Rendering**, *Signal Processing: Image Communication, Special Issue on Interactive Representation of Still and Dynamic Scenes*, Elsevier, vol 21, no 6, Oct, 2006, pp. 739-754. doi:10.1016/j.image.2006.08.002.
13. H. Tang, Z. Zhu, G. Wolberg and J. R. Layne, **Dynamic 3D Urban Scene Modeling Using Multiple Pushbroom Mosaics**, the *Third International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT 2006)*, University of North Carolina, Chapel Hill, USA, June 14-16, 2006.
14. Z. Zhu, H. Tang, **Content-Based Dynamic 3D Mosaics,** *IEEE Workshop on Three-Dimensional Cinematography (3DCINE'06),* June 22, 2006, New York City (in conjunction with CVPR'06)
15. Z. Zhu, H. Tang, G. Wolberg and J. R. Layne, **Content-Based 3D Mosaics for Dynamic Urban 3D Scenes**. *SPIE Defense and Security Symposium 2006*, 17 - 21 April 2006, Orlando, Florida, USA. **A feature article was selected into *SPIE Newsroom*,** doi:10.1117/2.1200607.0295

16. Z. Zhu, A. R. Hanson, **Mosaic-Based 3D Scene Representation and Rendering**, *Special Session on Interactive Representation of Still and Dynamic Scenes, the Eleventh International Conference on Image Processing*, Genova, Italy, September 11-14, 2005, pp I-633 -636

17. Z. Zhu, H. Tang, B. Shen, G. Wolberg, **3D and Moving Target Extraction from Dynamic Pushbroom Stereo Mosaics**, *IEEE Workshop on Advanced 3D Imaging for Safety and Security*, June 25, 2005, San Diego, CA, USA, http://doi.ieeecomputersociety.org/10.1109/CVPR.2005.376