Content-Based 3D Mosaics for Representing Videos of Dynamic Urban Scenes

Hao Tang, *Student Member, IEEE*, and Zhigang Zhu, *Senior Member, IEEE*

***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. 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 various real video sequences of large-scale 3D scenes.**

***Index Terms*— Multi-image registration, content-based video coding, image-based modeling, 3D scene representation**

I. Introduction

In this paper 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 amount of data is in the order of 100 GB per hour for standard 640*480 raw (uncompressed) color images. The huge amount of video data not only poses difficulties in data recording and archiving but is also 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 change detection algorithms that assume a 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, information such as the location of an abnormal event, the speed, flow and density information of the traffic of the entire area, can be immediately calculated and transmitted back to a control center, by using a fly though over an area. 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.

We propose a content-based 3D mosaic representation (CB3M) for long video sequences of such 3D and dynamic scenes. 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

Manuscript received October 17, 2007. This work is supported by: AFRL (Award No. FA8650-05-1-1853), NSF (Grant No. CNS-0551598), AFOSR (Award #FA9550-08-1-0199), ARO (Grant No W911NF-05-1-0011), and a PSC/CUNY Research Award. Some data were captured in the UMass Computer Vision Laboratory under an NSF grant (No EIA-9726401).

Hao Tang is with the Department of Computer Science, CUNY Graduate Center, New York, NY 10016, USA (e-mail: tang@cs.ccny.cuny.edu).

Zhigang Zhu is with the Department of Computer Science, CUNY City College, New York, NY 10031, USA (corresponding author; phone: 212-650-8799; fax: 212-650-6248;e-mail: zhu@cs.ccny.cuny.edu).

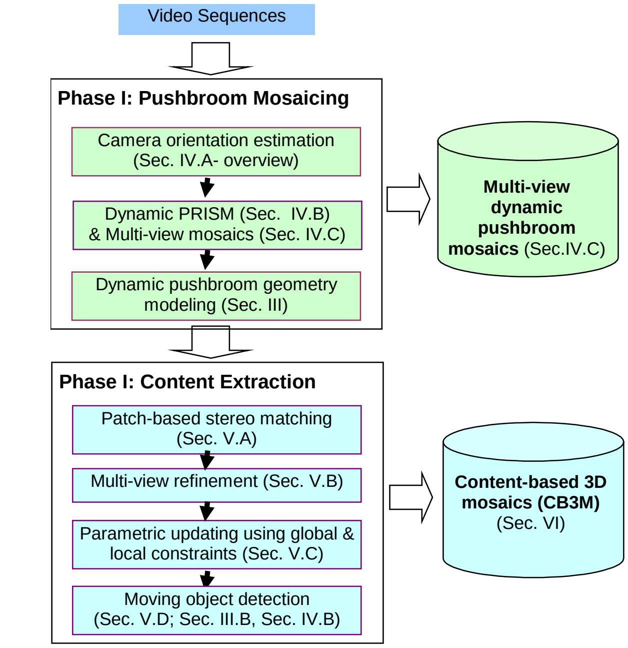

Video Sequences
Phase I: Pushbroom Mosaicing
Camera orientation estimation (Sec. IV.A- overview)
Dynamic PRISM (Sec. IV.B) & Multi-view mosaics (Sec. IV.C)
Dynamic pushbroom geometry modeling (Sec. III)
Multi-view dynamic pushbroom mosaics (Sec.IV.C)
Phase I: Content Extraction
Patch-based stereo matching (Sec. V.A)
Multi-view refinement (Sec. V.B)
Parametric updating using global & local constraints (Sec. V.C)
Moving object detection (Sec. V.D; Sec. III.B, Sec. IV.B)
Content-based 3D mosaics (CB3M) (Sec. VI)

Fig. 1. System diagram.

next phase.

However, the 2D mosaic representation is still an image-based one without object *content* representation. 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 of a dynamic 3D scene. 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. Therefore it 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 [1], we have proposed the parallel ray interpolation for stereo mosaicing (PRISM) algorithm that can generate seamless mosaics under motion parallax, for static scenes. In another piece of work [2], 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 [3].

Based on the previous work, we have made the following 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.

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. In addition, the new approach can also be used with other stereo geometry.

Finally, we propose a highly compact video representation for long video sequence of dynamic 3D scenes - the content-based 3D mosaic (CB3M) representation. We also perform thorough experimental analysis of the robustness, accuracy and efficiency of 3D reconstruction and representation using parallel-perspective stereo mosaics.

The rest of the paper is organized as follows. Section II discusses some related work. In Section III, the mathematical framework of the dynamic pushbroom stereo is given, and then its properties for moving target extraction are discussed. In Section IV, 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 V, our multi-view pushbroom stereo matching approach for 3D reconstruction and moving target extraction is provided. Then in Section VI, the content-based 3D mosaic representation is described. Experimental results of CB3M representation construction is given in Section VII. Section VIII gives concluding remarks and discusses some future research directions.

II. RELATED WORK

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 [4]-[7] 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 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 [1], [8]. The term "pushbroom" is borrowed from satellite pushbroom imaging [9] where a linear pushbroom camera is used. Pushbroom stereo mosaics have uniform depth resolution, which is better than with perspective stereo, and the multi-perspective stereo with circular projection [10], [11]. Pushbroom stereo mosaics can be used in applications where the motion of the camera has a dominant translational direction. Examples include satellite pushbroom imaging [9], airborne video surveillance [1], image-based rendering with 3D reconstruction or 3D estimation [8], [12], 3D representations of ground route scenes [13]-15], under-vehicle inspection [16], [17], 3D measurements of industrial parts by an X-ray scanning system [18], and 3D gamma-ray cargo inspection [19]. Some work has been done in 3D reconstruction of panoramic mosaics [20, [21] 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 [22]-24] 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 [25] has been performed. Simple window-based correlation approaches do not work well for man-made scenes. In the past, an adaptive window approach [26] and a nine-window approach [27] 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 [28], and for accurate layer extraction [22]. Global optimization based stereo matching methods, such as belief propagation [29] and graph cuts [30], [31], can obtain accurate depth information, but these methods are computationally expensive. A complete system was presented in [32] 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 [33] 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 [14],[20],[21] dealing with multi-perspective stereo images.

III. 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 [9]. Therefore we also call this image representation *pushbroom stereo mosaic representation*. A generalized model under 3D translation [1] 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.

A. 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

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

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 (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 pixels) 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. 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 [2]. 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

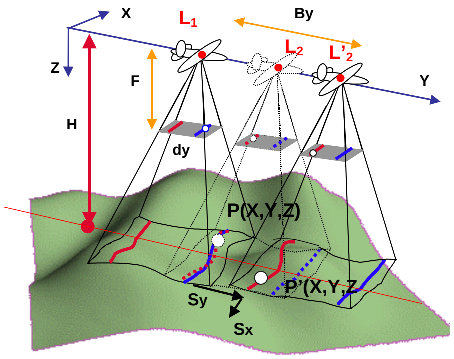


Fig. 2. Dynamic pushbroom stereo mosaics

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

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 in (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 [1].

B. Moving Object Extraction against Parallax

We have made 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.

(i) *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.

(ii) *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).

(iii) *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.

(iv) *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 (5), and (S_x, S_y) using (6), knowing the visual motion (Δx, Δy) measured in the stereo mosaics.

IV. Real-World Issues and Multi-View Mosaics

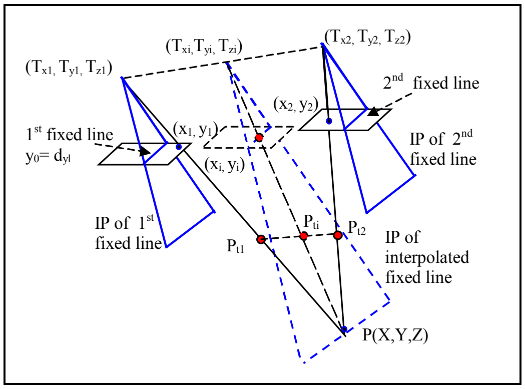
In real applications, there are three sets of challenging problems. These include camera motion estimation in practical cases, mosaic generation with more general camera motion, and occlusion and stereo matching issues in a pair of stereo mosaics. For some issues, we will give very brief discussions 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.

A. 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 [1]-[3]. More general approaches are bundle adjustment techniques [34] 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. 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, (i) 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 (ii) 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 a 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.

B. 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 [1] 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.



(T$_{x1}$, T$_{y1}$, T$_{z1}$)
(T$_{xi}$,T$_{yi}$, T$_{zi}$)
(T$_{x2}$, T$_{y2}$, T$_{z2}$)
2nd fixed line
(x$_2$, y$_2$)
(x$_1$, y$_1$)
1st fixed line
y$_0$= d$_{yl}$
(x$_i$, y$_i$)
IP of 2nd fixed line
IP of 1st fixed line
P$_{ti}$
P$_{t2}$
P$_{t1}$
IP of interpolated fixed line
P(X,Y,Z)

Fig. 3. Ray interpolation for a dynamic scene

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

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

(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 linear pushbroom scan of the moving point. Under the linear motion assumption, the mosaic coordinates of the pair of point are

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

This is an important finding since the mosaicing algorithms developed for static scenes can be directly applied to dynamic scenes. 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 [1] has been designed, 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.

C. *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. Second, it eases the stereo correspondence problem in a way similar to multi-baseline stereo [35], 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. In the next section, we will discuss in details a new method to extract both 3D structures and moving targets from multiple dynamic pushbroom mosaics. We will also discuss the possibility of extracting and representing occluded regions in Section VI.

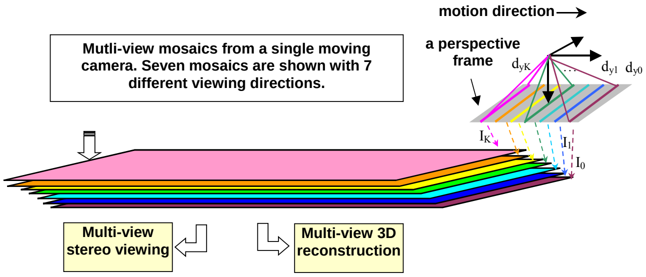

motion direction
Mutli-view mosaics from a single moving camera. Seven mosaics are shown with 7 different viewing directions.
a perspective frame
d_{yK}
d_{y1}
d_{y0}
I_K
I_1
I_0
Multi-view stereo viewing
Multi-view 3D reconstruction

Fig. 4. Multi-view pushbroom mosaics

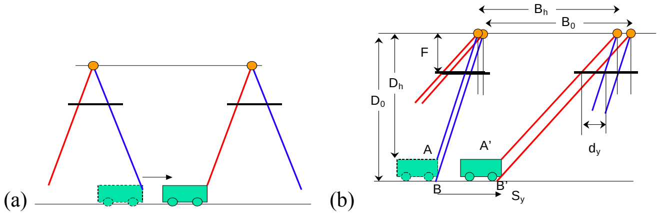


Fig. 5. Height from dynamic stereo: (a) an infeasible pair; (b) a feasible pair

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 the 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 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. 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 (10). Finally, the depth of the point on the top, Z_h, can be estimated using (9), given the known visual motion of that point, Δy_h, and its independent motion component s_y obtained from the bottom point *B*.

V. 3D and Motion Content Extraction

Using the advantageous properties of multi-view mosaics, we propose a unified approach that consists of four stages to perform both stereo matching and motion detection. First, 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. Second, 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. Third, local and global spatial constraints are also explored to improve the robustness of the 3D estimation. Finally, moving targets are detected after the "3D alignments" of the scene.

A. 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 [36] 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 on the boundary 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 there are 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:

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

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

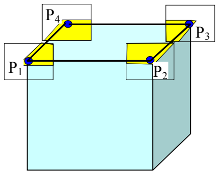


Fig. 6. Natural matching primitives

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

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

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 [27]), 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 (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, and a new set of parameters (S_h, S_v and *s*) are calculated based on the first run (i.e., the search range is narrowed to neighborhood of corresponding point with a finer step, therefore S_h, S_v and *s* are reduced). Then, the natural matching 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

$$a_i X + b_i Y + c_i Z = d_i \tag{14}$$

which is represented in the camera coordinate system as shown in Fig. 2. A 3D plane is fitted to each region after obtaining the 3D coordinates of the interest points of the region using the pushbroom stereo geometry (in (1) and (2)).

We use a standard RANSAC method [37] 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 a supporter to the fitted plane. The total number of supporters 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 supporters 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 an unreliable patch at this point is the one whose number of reliable interest points is smaller than 3, or the total number of the plane supporters does not exceed the required percentage (i.e. 65% in our experiments). In the end, there are three categories of patches: those with a 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

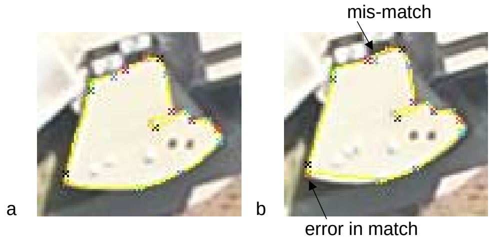


Fig. 7. An example of region matching results. The matches are marked as "X", with corresponding colors.

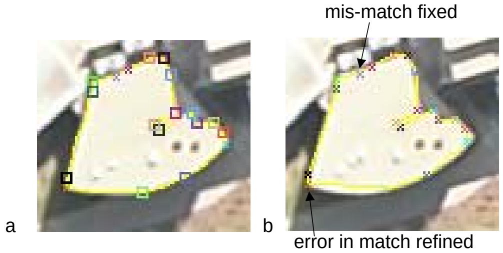


Fig. 8. An example of surface fitting results. Both the mismatch and the small error in the initial match are fixed.

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

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.

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 (cropped from the reference and target mosaics), 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.

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.

B. 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 (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 V.A 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 $\mathbf{\Theta}_{ik} = (a_{ik}, b_{ik}, c_{ik}, d_{ik}), k=1,\ldots,K$, are obtained for each region (patch) in the reference mosaic (some regions have fewer than K sets of available plane parameters due to insufficient 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 (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 (18), with k =0. Then again, using (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 (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.

C. Plane Updating Using Local and Global 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 greater than a preset threshold T_i, then the patch is moved to the *unreliable* category (C_i = 1) under plane fitting, multi-view refinement and SSD evaluation, therefore the attributes in (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 the threshold T_i is defined as

$$T_i = Q_i \times 3 \times D^2 \tag{21}$$

where 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 some small regions around larger regions, 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.

Neighbor patch supporting

In the neighborhood supporting strategy, we perform a modified version of the neighboring plane parameter hypothesis algorithm [30] to infer better plane estimates. Based on our region categorization, two main modifications are made: (a) the parameters of a neighboring region are adopted only if the region is marked reliable; and (b) 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.

(i). 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,…J } for the current region $\mathbf{R}_i$, including the current region, therefore M<= J+1.

(ii). Apply the parameter set Θ_{jm} (m=1, 2, …M) to the region $\mathbf{R}_i$, to calculate the corresponding $SSD_{i,jm}$(m=1, 2, …M), using (20).

(iii). Select the parameter set Θ_{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 share the same plane parameters, then they are then merged into one reliable region. This step is performed recursively until no more merges occur. We prefer to have false negatives than false positives, and the former will be handled in the next stage – moving object detection.

Global scene constraints

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.

(i) 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.

(ii) 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 VII, with both simulated and real video sequences.

D. 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 III.B, observation (iv)). 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 IV 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: (i) moving objects with motion not obeying the pushbroom epipolar geometry; (ii) 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 processed properly 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. Only if the region size is less than 300 pixels, it goes through the moving target detection procedure.

The moving target detection steps are summarized as follows.

(i) For all reliable regions with less than 300 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: the height of the region $\mathbf{R}_i$ is 20 meters higher than the average height of the neighboring regions $\{\mathbf{R}_{ij}\}$; or the height of the region $\mathbf{R}_i$ is 10 meters lower than the average height of the neighboring regions.

(ii) For *all* unreliable regions with less than 300 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:

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

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 (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).

VI. 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.

A. 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 (22). As a summary, they are explained below: (i) N is the number of VOs, i.e., "homogeneous" color patches (regions); (ii) $\mathbf{c}_i$ is the color (3 bytes) of the *ith* region; (iii) $\mathbf{b}_i$ 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) uses 4 bytes, and each chain code uses 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; (iv) $\{\mathbf{R}_{ij}$, j =1,…, $J_i\}$ 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. $J = (1/N) \sum J_i$); (v) $C_i = 2$, if the region is a static patch with reliable plane parameters (see (vi)); $C_i = 1$, if the region is a moving target (therefore with **m**i, see (vii)); $C_i = 0$, otherwise (unreliable, maybe occluded regions). (vi) $\mathbf{\Theta_i} = (a_i, b_i, c_i, d_i)$ represents the plane parameters of the region in 3D, 4 bytes for each parameter; and (vii) $\mathbf{m_i}$ 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)

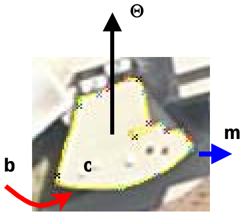


Fig. 9. Content-based 3D mosaic representation.

$$\begin{aligned}&N_{color} + N_{boundary} + N_{neighbor} + N_{structure} + N_{motion} \\ &= 3N + (8N+3G/8) + 4JN + 4*4N + 4M*N_m \\ &= (27+4J)N + 3G/8 + 4MN_m \text{ (bytes)}\end{aligned} \tag{24}$$

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 CB3M 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. 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.

B. Representing Occlusion and Higher Level Objects

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 V 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 [14]. In the end, the extended CB3M representation will have the following three components:(i) A base layer that consists of a set of planar patches corresponding to the reference mosaic; (ii) 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 (iii) 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 maps, and can 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 (e.g., vehicles, buildings, roads, humans), is also possible, for applications such as automated target recognition and 3D model indexing.

VII. Experimental Results and Analysis

The proposed approach for the content-based 3D mosaic representations was applied to multi-view pushbroom mosaics generated from both simulated and real world video sequences of both indoor and outdoor scenes. With simulated video scenes, we have performed evaluations on the accuracy of 3D and motion estimation using multiple pushbroom mosaics, with ground truth data [38]. Here we only present two real-world examples: the flyover of a campus scene, and the flyover of a New York City (NYC) scene. For more experimental results and more detailed analysis, please refer to [38]. Finally, we will provide some analysis on computation time in both stereo mosaicing and content extraction.

A. Results on a Campus Scene

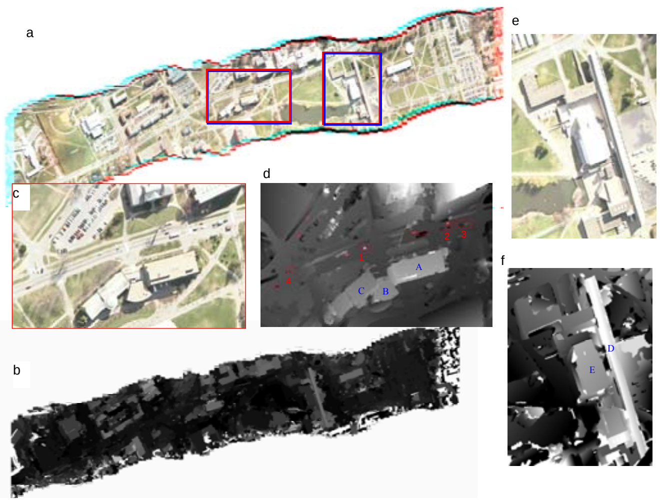


Fig. 10. 3D and motion from multi-view stereo mosaics of an aerial video sequence. (a) A pair of stereo mosaics from the total nine mosaics; (b) height map of entire mosaic; (c) close-up of the 1st window marked in (a); and (d) the height map of the objects inside that window, with the detected moving 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.

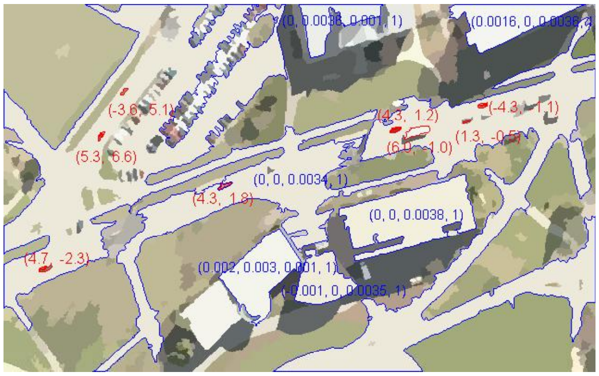


Fig. 11. Content-based 3D mosaic representation of an aerial video sequence. Only a window is shown, with some of the regions labeled by their boundaries and plane parameters (*a,b,c,d*), and the detected moving targets marked by their boundaries and motion vectors (s_x,s_y).

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. 10a 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. 10b is the "height" map (corresponding to the reference mosaic) using the proposed method. Fig. 10c and Fig. 10d, Fig. 10e and Fig. 10f show the images of the two close-up windows and the corresponding "height" maps. Note that 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. 10d and Fig. 10f 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).

The CB3M mosaic (of the first window in Fig. 10a) is shown in Fig. 11, with a color value, a boundary, plane parameters and a motion vector (if in motion) for each patch (region). 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 raw data amount is 879 MB. The size of a pair of stereo mosaics (Fig. 10a) is 4448*1616*2, which uses 41MB of storage (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 are 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 (24), which is about 315 KB. The data amount is reduced to 90 KB with a simple lossless WinZip compression on the CB3M data; therefore, the compression ratio is about **10,001**. Note that the CB3M representation in Fig. 11 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.

B. *Results on an NYC Scene*

The NYC mosaics were generated from a video sequence from an NYC HD (high-definition) aerial video dataset 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 there are 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 of various heights, from less than ten to more than a hundred meters. Fig. 12 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. 13.

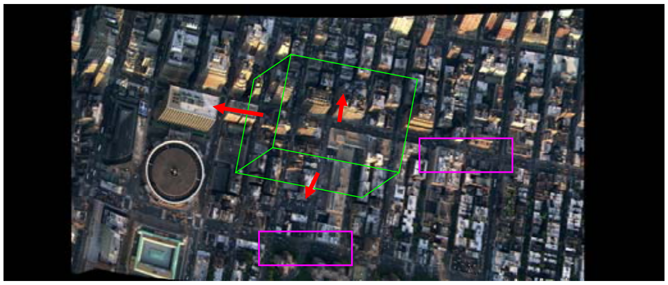


Fig. 12. A 4816 (W) x 2016 (H) mosaic from a 758-frame high-resolution NYC video sequence. The Manhattan world geometric constraint is illustrated on the mosaic. The two rectangular windows include cars running in two one-way roads, respectively.

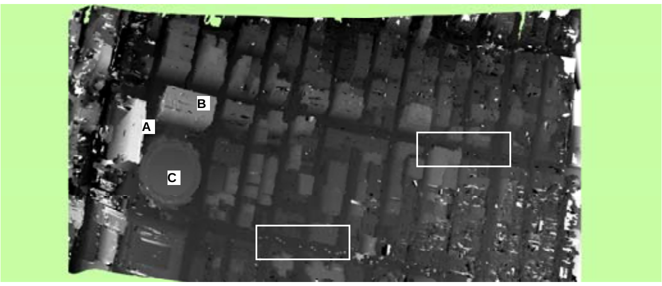


Fig. 13. Depth from four pushbroom mosaics. Within the two windows. Moving targets are detected as “outliers”.

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 [39] to further refine the 3D reconstruction results. As shown in Fig. 12, 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 all of the 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. For this, the two-step strategy in using the global scene constraints discussed in Section V.C is applied. Then, the rest of the regions, i.e. the “outliers”, go through the moving object detection test. We use the same method as presented in Section V.D, 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 rooftops).

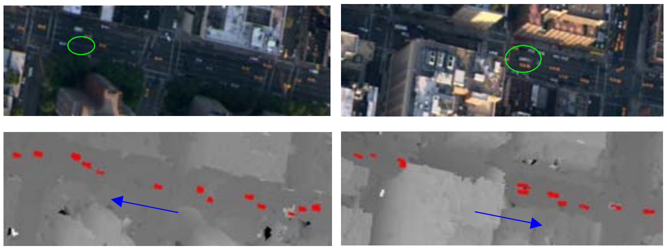


Fig. 14. Moving target detection using the road direction constraint. In the figure (a) and (b) are the corresponding color images and height maps of the 1st (bottom-left) and 2nd (top-right) windows in Fig. 12, with the detected moving targets painted in red. The two circles show the three moving targets that are not detected. The arrows indicate the directions of the roads along which the moving targets are searched.

Fig. 13 shows the 3D reconstruction results of the NYC video data, represented in the leftmost mosaic - the reference mosaic. The figure shows the height map generated from multi-view mosaics. For the improvements of depth estimation using multiple views over two views, please refer to [38]. Due to the lack of flight and camera parameters, we roughly estimate the main parameters of the camera (i.e., the height of the flight and the camera's focal length) 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. 13) 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.

The moving objects (vehicles) create "outliers" in the height map, as can be clearly seen on the height map (the brighter the color is, the higher the object is). For example, on the one-way road indicated in the first window in Fig. 12, vehicles moved from the right to the left in the figure, therefore, their 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. 12, vehicles moved from the left to the right in the figure, therefore, their estimated heights are much lower than the ground if assumed static). After further applying the knowledge of road directions that are obtained from a dominant plane clustering procedure, moving targets are searched and extracted. In Fig. 14, 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 were parked on these two one-way roads.

C. Computation Time Analysis

The two-phase CB3M construction is also efficient in computation time. The following statistics were 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 [3].

TABLE I.
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
NYC	3700x2000	37166	3	(30, 8)	330	8.88	9420	84.49

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 following step of matching multi-view pushbroom mosaics). The segmentation step was implemented using the mean shift algorithm [36] and a toolbox provided by the authors, and the matching step was implemented by us in C++. Table I lists the time performance for the two video sequences presented in this paper: the campus 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 matching (Section V.A), and image warping in match evaluation (i.e., SSD) in the multi-view refinement (Section V.B) and plane updating using global and local constraints (Section V.C). 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 I 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 two 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.

VIII. Concluding Remarks

In this paper 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 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. 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 paper. First, in the CB3M representation presented in this paper many details and practical issues have not been considered. 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. 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 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 IV should be applied. This also requires further analysis of relations of object regions (patches).

Second, we would like to generalize the pushbroom stereo mosaicing approach with more general camera motion. For example, we are working on stereo mosaicing with circular camera motion, and we have derived a geometric model for such a case. In the long term, we would like to combine pushbroom stereo mosaicing techniques in linear and circular motion cases, and generalize them to situation with more a general camera motion path. 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.

Acknowledgment

The authors want to thank Edgardo Molina and Prof. George Wolberg at CCNY, Dr. Jeff Layne, Dr. Greg Arnold and Ms. Olga Mendoza at AFRL for their valuable discussions during the course of the related projects. We also thank anonymous reviewers for their constructive comments and suggests that help to greatly improve the presentation of the paper.

References

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

[2] Z. Zhu, A. R. Hanson, H. Schultz and E. M. Riseman, 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 2003: 72-105

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

[18] A. Noble, R. Hartley, J. Mundy and J. Farley, X-Ray metrology for quality assurance, In *Proc. IEEE Int. Conf Robotics and Automation (ICRA)*, 1994, pp 1113-1119

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

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

[21] C. Sun and S. Peleg, Fast panoramic stereo matching using cylindrical maximum surfaces, *IEEE Trans. System, Man and Cybernetics*, Part B, 34, Feb. 2004: 760-765.

[22] Q. Ke and T. Kanade, A subspace approach to layer extraction, In *Proc. Computer Vision and Pattern Recognition (CVPR).* 2001.

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

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

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

[26] T. Kanade and M. Okutomi, A stereo matching algorithm with an adaptive window: theory and experiment, In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 1991, II: 1088-1095

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

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

[29] J. Sun, N. Zheng and H.-Y. Shum, Stereo matching using belief propagation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(7), July 2003

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

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

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

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

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

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

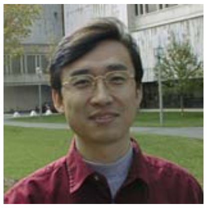
[36] D. Comanicu and P. Meer, Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Patten Analysis and Machine Intelligence*, May 2002

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

[38] H. Tang and Z. Zhu, Content-based 3D mosaics for large-scale dynamic urban scenes. *Technical Report TR-2008013*, Computer Science Department, Graduate Center, City University of New York, September 2008. http://tr.cs.gc.cuny.edu/tr/techreport.php?id=364

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

Hao Tang received his B.S. degree from Beijing Polytechnic University, Beijing, China, in 1992, and the M.S. degree from the City College of New York in 2003, both in computer science. He is now a Ph.D. candidate at the Graduate Center of the City University of New York. Since 2003, he has been a research assistant in the City College Visual Computing Laboratory, working on video surveillance and 3D computer vision. He is a student member of IEEE.



Zhigang Zhu received his B.E., M.E. and Ph.D. degrees, all in computer science from Tsinghua University, Beijing, China, in 1988, 1991 and 1997, respectively. He is currently a full professor in the Department of Computer Sciences, the City College of the City University of New York. He is Director of the City College Visual Computing Laboratory (CcvcL), and Co-Director of the Center for Perceptual Robotics, Intelligent Sensors and Machines (PRISM) at CCNY. Previously he has been Associate Professor at Tsinghua University, and Senior Research Fellow at the University of Massachusetts, Amherst. His research interests include 3D computer vision, Human-Computer Interaction (HCI), virtual / augmented reality, video representation, and various applications in education, environment, robotics, surveillance and transportation. He has published over 100 technical papers in the related fields. He is a senior member of the IEEE, a senior member of the ACM and an associate editor of the Machine Vision Applications Journal.