Chapter 2.
Panoramic Vision for Landmark Recognition

Abstract

This chapter presents a systematic approach for automatically constructing a 3D panoramic model of a natural scene from a video sequence for landmark localization of a mobile robot in an outdoor road scene. The video sequences could be captured by an unstabilized camera mounted on a moving platform on a common road surface. First, a 3D image stabilization method is proposed which eliminates fluctuation from vehicle's motion so that "seamless" panoramic view images (PVIs) and epipolar plane images (EPIs) can be generated. Second, a comprehensive panoramic EPI analysis method is proposed to combine the advantages of both PVIs and EPIs efficiently in two important steps: locus orientation detection in the frequency domain, and motion boundary localization in the spatio-temporal domain. Finally, The texture map and the depth map of the route-based panoramic view representation are used to extract landmarks for mobile robot navigation. Since camera calibration, image segmentation, feature extraction and matching are avoided, all the proposed algorithms are fully automatic and rather general. Results of image stabilization and 3D construction for real image sequences are given.

2.1. Introduction

What do we memorize when we drive in an unknown urban scene, such as a university campus or a downtown area? We use landmarks (both natural ones and artificial ones) and their spatio-contexts. We cannot build an exact 3D model (with texture mapping) of the scene. Instead we may maintain a visual map based on landmarks of the road network that locate along the road side. Without doubt, 3D information is also useful. But how to represent these landmarks of the 3D scene in our memories? If we want to have a mobile robot (autonomous vehicle) to do this, the most suitable sensor is a video camera. Hence the task turns to be maintaining a suitable visual representation of a large-scale 3D scene, and in order to do this, the essential issue is motion analysis of a long video sequence.

In this chapter, we will address the problem of automatically constructing a 3D panoramic model of a static natural scene from an easily-obtained video sequence. We do not attempt to solve the general structure from motion problem; instead, the motion of the camera is somewhat constrained. That is, we assume that a dense image sequence can be captured by an uncalibrated camera mounted on an ordinary vehicle, moving on a common (and often bumpy) road surface. Accurate motion parameters are generally unknown; the only thing known is that the camera roughly points perpendicular to the motion direction and that it is subject to an uncontrollable fluctuation. No assumption is made on the structure of the scene. The goal is to construct a compact representation of a large scale 3D scene from ordinary video sequences. To begin with, a multi-perspective panoramic view image (PVI) and a set of epipolar plane images (EPIs) are extracted from a long image sequence captured in the manner described above, and then a depth value is calculated for each pixel of the panoramic image by analyzing the corresponding EPIs. Thus we try to solve three problems: (1) how to generate seamless PVIs and EPIs from video under a more general motion than a pure translation; (2) how to analyze the huge amount of data in EPIs robustly and efficiently; and (3) how to extract landmarks from the texture and 3D maps.

It has been shown that under strict translation, a panoramic view image (PVI) can be generated by extracting a vertical column from each frame and piling them up to form a wide angle multi-perspective image (Zheng and Tsuji, 1992), and it has been used in mobile robot navigation. Similarly, a well-known epipolar plane image (EPI) first proposed by Bolles et al. (1987) can be generated by extracting a horizontal scan-line from each frame and piling them up to form a spatio-temporal (ST) image, where the orientation angle of a locus is proportional to the depth of the corresponding point. Techniques based on 2D ST image formation (panoramic view images and epipolar plane images) meet the need for a compact representation and fast 3D recovery (e.g., Ishiguro et al., 1990; Zheng and Tsuji, 1992; McMillan and Bishop, 1995; Dalmia and Trivedi, 1996; Murray, 1995; Shum and Szeliski, 1999); however the strong constraints of pure translation (or perfect rotation) limit the wide use of PVI representations and EPI methods. For EPI-based depth recovery methods, locus extraction is also a hard problem for image sequence of a natural scene with complex textures and unpredictable bumping in camera motion. In addition, the large amount of data in EPIs often makes it prohibitive in computation.

In order to apply these two kinds of compact representations to an easily-captured image sequence, we have proposed and implemented a two-stage approach. In the first stage, a 3D image stabilization method is proposed to de-couple vibrating motion and scene structure, thus making the epipolar plane image analysis and the panoramic view image representation valid for unstabilized image sequences. Notice that a commercial off-the-shelf camcorder with a digital stabilizing function usually distorts the perspective geometry of an image sequence because it uses 2D translation to stabilize the video sequence. Existing digital stabilization algorithms (e.g., Hansen, et al., 1994; Morimoto and Chellappa, 1997) are not designed to meet the need of keeping the loci straight in an EPI for a long image sequence. Our 3D stabilization algorithms with 3D image warping and motion filtering are specially designed for EPI generation and 3D recovery. Three algorithms are proposed for motion filtering, namely the locus tracking and fitting approach, motion classification and selection approach and statistical locus smoothing/fitting approach. Experimental results on many real image sequences have shown that seamless PVI and EPI can be generated.

In the second stage, a Fourier energy method over a large Gaussian-windowed area of an EPI is proposed to robustly detect multiple orientations of the EPI's motion texture in the frequency domain. This approach is different from the commonly-used locus tracking method (Murray, 1995; Allmen and Dyer, 1991), or local operator methods, such as Gabor filters (Adelson and

Bergen, 1985; Heeger, 1987) and Steerable filters (Freeman and Adelson, 1991; Niyogi, 1995; Fleet et al., 1998), where only limited angle resolution can be obtained since a local motion detector is often performed in a small ST neighborhood. As far as we know, our work seems to be the first attempt to use large neighborhood windows (e.g. 64×64) for detecting local motion more robustly and accurately. Furthermore, motion boundaries are accurately located back in the spatio-temporal domain by measuring global intensity similarities only along the detected orientations. Occluded regions can be recovered by further exploring extra information near motion boundaries in the EPI[1]. Three-dimensional panoramic models have been constructed from several image sequences, some of which have more than 1000 frames. Most significantly, direct methods for all the steps have been developed in which image segmentation, feature extraction, and matching are avoided. We emphasize that only a small number of selected data corresponding to the selected PVI is processed in our approach, and the processing for all the epipolar planes can be done in parallel. Thus it is quite possible to implement the proposed algorithms in real time. Even the current sequential implementation in a Pentium 400 MHz PC can achieve a frame rate of about 2 frame per second for 128×128 images. For robot navigation application, sparse (or low resolution) 3D estimation may be enough for effectively extracting landmarks for robot localization, so it makes the realtime implementation even feasible.

The rest of this chapter is organized as follows: Section 2 describes the principle and algorithms of image stabilization for generating panoramic view images and epipolar plane images. In section 3, a motion occlusion model is presented first, and then a Gaussian-windowed Fourier method is proposed for multiple-motion orientation detection. Three key algorithms will be presented in this section for panoramic epipolar plane image analysis that leads to a dense depth map with accurate depth boundary localization. In Section 4, an integrated system of constructing 3D panoramic model will be described, and the issues on landmark selection and robot localization will be briefly discussed. Experimental results of several image sequences for natural scene modeling and rendering are provided. A brief conclusion and discussion are given in the last section.

2.2. Motion Filtering and Image Stabilization

2.2.1. The Vehicular Motion Model

Suppose that a camera is mounted on a vehicle moving on an approximately flat road surface. In order to construct the 3D model of a roadside scene, the camera's optical axis is perpendicular to the motion direction and its horizontal axis is parallel to the motion direction[1]. Within a considered long time period $[0,T]$, we assume that the motion of the vehicle (camera) consists of a smooth planar motion and an unpredictable small fluctuation due to the vehicle's motion over a rough surface. In many real cases, the smooth motion can be approximated as a constant velocity (V) translation. The small fluctuation between two successive frames is modeled by three small rotation angles $\Omega_x, \Omega_y, \Omega_z$ around the X, Y and Z axes and three translation components T_x, T_y, T_z along the three axes (Fig. 1). A generalization of this model is given in Appendix 2.1 where the vehicle can move along a curved path, and yet the proposed methods in this chapter are still valid.

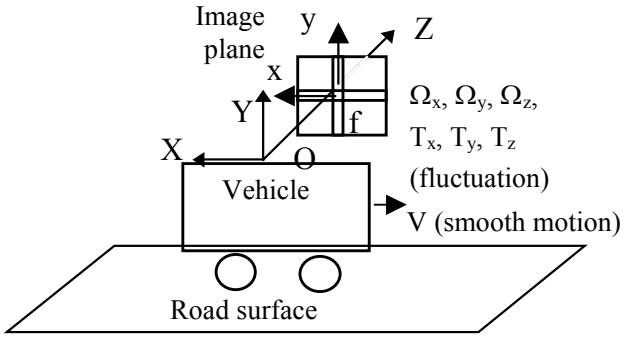


Fig. 1. The vehicular motion model

Under this assumption, the relationship between coordinates of a 3D point at time t and time t-1 can be expressed as

$$\begin{pmatrix} X_{t-1} \\ Y_{t-1} \\ Z_{t-1} \end{pmatrix} = \begin{pmatrix} 1 & \Omega_z & -\Omega_y \\ -\Omega_z & 1 & \Omega_x \\ \Omega_y & -\Omega_x & 1 \end{pmatrix} \begin{pmatrix} X_t \\ Y_t \\ Z_t \end{pmatrix} + \begin{pmatrix} T_x - V \\ T_y \\ T_z \end{pmatrix}$$

Using a pinhole camera model $(x,y) = \left(f\frac{X}{Z}, f\frac{Y}{Z} \right)$, the relation of the image coordinates in the sensing targets of the two successive images under small motion can be represented as (Sawhney and Ayer, 1996; Black and Jepson, 1996)

$$s\begin{pmatrix} x_{t-1} \\ y_{t-1} \end{pmatrix} = \begin{pmatrix} x_t \\ y_t \end{pmatrix} + \begin{pmatrix} -\frac{x_t y_t}{f} & \frac{{x_t}^2 + f^2}{f} & -y_t \\ -\frac{{y_t}^2 + f^2}{f} & \frac{x_t y_t}{f} & x_t \end{pmatrix} \begin{pmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{pmatrix} + \frac{1}{Z} \begin{pmatrix} f & 0 & -x_t \\ 0 & f & -y_t \end{pmatrix} \begin{pmatrix} T_x - V \\ T_y \\ T_z \end{pmatrix} \tag{1}$$

where s is a zooming factor between two image frames. The relation between the image coordinate *(x, y)* (in mm) and the digital frame coordinate *(u, v)* (in pixels) can be expressed as

$$(x, y) = (f\, s_x u, f\, s_y v) \tag{2}$$

where s_x/s_y is the aspect ratio of the image sensor considering the effective focal length in pixels. Then the relation between frame coordinates in time *t* and *t-1* can be derived as

$$\begin{pmatrix} u_{t-1} \\ v_{t-1} \end{pmatrix} = \begin{pmatrix} u_t \\ v_t \end{pmatrix} + \begin{pmatrix} a & b & c & g & h & 0 \\ d & e & b & 0 & g & h \end{pmatrix} \begin{pmatrix} 1 \\ u_t \\ v_t \\ {u_t}^2 \\ u_t v_t \\ {v_t}^2 \end{pmatrix} + \frac{1}{Z} \begin{pmatrix} 1 & 0 & -u_t \\ 0 & 1 & -v_t \end{pmatrix} \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \tag{3}$$

where

$$\begin{cases} a = \frac{1}{s_x}\Omega_y,\ b = \frac{1}{s} - 1,\ c = \frac{-s_y}{s_x s}\Omega_z,\ d = -\frac{1}{s_y}\Omega_x,\ e = \frac{s_x}{s_y s}\Omega_z, \\ g = \frac{s_x}{s}\Omega_y,\ h = -s_y\Omega_x,\ t_x = \frac{T_x - V}{s_x s},\ t_y = \frac{T_y}{s_y s},\ t_z = \frac{T_z}{s} \end{cases}$$

Given *N* pairs of points $(u_{t,i}, v_{t,i})$ and $(u_{t-1,i}, v_{t-1,i})$ in frames *t* and *t-1*, $(i=1,...,N)$, we will have *2N* equations from Eq. (3), with *N+6* unknown parameters $(a_1, \ldots a_N, b', c, d', e, g, h)$ in the case of small translational components (T_x, T_y, T_z) , i.e.

$$\begin{cases} u_{t-1,i} = u_{t,i} + a_i + b' u_{t,i} + c v_{t,i} + g u_{t,i}^2 + h u_{t,i} v_{t,i} \\ v_{t-1,i} = v_{t,i} + d' + e u_{t,i} + b' v_{t,i} + g u_{t,i} v_{t,i} + h v_{t,i}^2 \end{cases} \tag{4}$$

where

$$a_i = \frac{1}{s_x}\left(\Omega_y + \frac{T_x}{s Z_i}\right) - \frac{V}{s_x s Z_i}\ ,\ b' \overset{\Delta}{=} b_i = \frac{1}{s}\left(1 - \frac{T_z}{Z_i}\right) - 1 \tag{5}$$

$$d' \overset{\Delta}{=} d_i = \frac{1}{s_y}\left(-\Omega_x + \frac{T_y}{sZ_i}\right),\ c = \frac{-s_y}{s_x s}\Omega_z,\ e = \frac{s_x}{s_y s}\Omega_z,\ g = \frac{s_x}{s}\Omega_y,\ h = -s_y\Omega_x$$

In Eqs. (4) and (5), b' and d' are approximately constant values over the entire image in time t when (T_x , T_y , T_z) are very small and/or the depths of all the *selected* points do not vary too much. (Point pairs can be selected based on the closeness of their displacement ($u_{t,i}$-$u_{t-1,i}$, $v_{t,i}$-$v_{t-1,i}$)). Eq. (4) can be solved by giving more than 6 point pairs ($N \geq 6$). A hierarchical block matching and motion estimation algorithm (see Appendix 2.2; see also Zhu et al., 1999c) has been used to estimate the 6+N unknowns in Eq. (4).

The image stabilization in the following is a process of eliminating fluctuations of the vehicle so that the motion after stabilization is a translation motion with constant velocity V within the time period $[0,T]$. This is the basic difference between our 3D image stabilization method and other image stabilization methods (e.g., Hansen, et al, 1994; Morimoto and Chellappa, 1997). The proposed 3D image stabilization approach consists of two steps: image rectification between the current frame and the previous *rectified* frame, and motion filtering over the entire image sequence.

2.2.2. Image Rectification

The aim of image rectification is to generate an image sequence that has horizontally parallel motion parallax between each pair of successive frames. From Eq. (5) we can define

$$a_i \overset{\Delta}{=} a' - k^{(i)}V \tag{6}$$

where

$a' = \frac{1}{s_x}\left(\Omega_y + \frac{T_x}{sZ_i}\right)$ is the fluctuating component of motion in the x direction, and is approximated as a constant parameter for all points.

$k^{(i)} = \frac{1}{s_x s Z_i}$ is defined as the "*projective depth*" of point i.

Unfortunately we cannot find a' directly from Eq. (6) since we do not know $k^{(i)}V$ in advance. However, a' can be roughly estimated by using the relation of a', a_i and g in Eq. (5) and (6), as

$$a' = \frac{s}{s_x^2} g \tag{7}$$

Note that the effect of the translational part T_x/sZ_i is ignored in Eq. (7), which will be re-considered and compensated in the motion filtering step. Using *a'* calculated from Eq. (7) instead of the actual *a'* will still yield a rectified image, since a_i only causes a horizontal shift (Eq. (4)). Thus frame *t* can be warped to a rectified image (u'_t, v'_t) by using the following equation

$$\begin{cases} u'_t &= u_t + a' + b'u_t + cv_t + gu_t^2 + hu_t v_t \\ v'_t &= v_t + d' + eu_t + b'v_t + gu_t v_t + hv_t^2 \end{cases} \tag{8}$$

By denoting $\mathbf{u}_t = (u_t, v_t)$ and $\mathbf{u}'_t = (u'_t, v'_t)$ as the coordinates of a point in the original and warped images, and the warping function as $\mathbf{W_t} = \{(a',b',c,d',e,g,h)\}$, the image rectification in Eq. (8) can be expressed as

$$\mathbf{u}'_t = \mathbf{W_t}(\mathbf{u_t}) \tag{9}$$

Hence, an estimation of motion parallax ($k^{(i)}V$) of point *i* due to smooth motion *V* can be derived as

$$\hat{m}_t^{(i)} = -a_i + a' \tag{10}$$

where a_i is estimated in Eq. (4) and a' is estimated in Eq. (7). Now we will summarize the rectification process of a video sequence. Naturally the first frame (frame 0) of a image sequence is used as the reference frame. It should be warped to a rectified image using some kind of offline calibration (see Section 4.2), so that the *x* axis of the first frame will coincide with the motion direction of the camera, i.e.

$$\mathbf{u}'_0 = \mathbf{W}_0(\mathbf{u}_0) \tag{11}$$

where $\mathbf{u}'_0$ is in the desired coordinate system. The remaining frames are rectified in the following manner. Frame *t* is matched with the *warped image* of frame *t-1*. Then the warping function $\mathbf{W}_t$ for frame *t* can be found by using Eq. (4) and Eq.(7), where ($u_{t-1,i}$, $v_{t-1,i}$) is replaced by warped coordinates $\mathbf{u}'^{(i)}_{t-1} = (u'_{t-1,i}, v'_{t-1,i})$, and an estimation of motion parallax $\hat{m}_t^{(i)}$ for each point *i* is computed by Eq.(10). After image rectification we have

$$\mathbf{u}'^{(i)}_{t-1} = \mathbf{u}'^{(i)}_{t} - \mathbf{m}^{(i)}_{t} \tag{12}$$

where $\mathbf{m}^{(i)}_{t} = (\hat{m}^{(i)}_{t},0)^{T}$ is the motion parallax of point *i* between the current frame *t* and the previous frame *t-1*, both of them are warped. Obviously, the interframe motion parallax field becomes a horizontal parallel field in the *x* direction after image warping. Note that in the above procedures of match and estimation, there is no requirement to track any point *i* across multiple frames. Index *i* is only valid between a pair of successive frames.

2.2.3. Motion Filtering Algorithms

Even if the motion parallax field becomes a horizontal parallel field in the *x* direction between two successive frames, the locus of a feature point across multiple frame may not be a straight line due to motion model approximation (Eq.(4)) and the decomposition in Eqs. (7) and Eq. (10). Note that $\mathbf{m}^{(i)}_{t}$ can also form a parallel field by adding any constant value (instead of a') to the motion parallax of each point *i*. To remove this ambiguity, in principle, we need to assume that a specific feature point *i* can be tracked across multiple frames ($t=0,1,...,T$) in the image sequence. Then the task of motion filtering is to find a straight line out of point set $\{(\hat{m}^{(i)}_{t}, t)\ (t=0,1,...,T)\}$, which satisfies the assumption of a 1D translational motion with the constant speed *V*. The procedure of the motion filtering is to find a small shift q_t for each frame so that the shifted image

$$\hat{u}^{(i)}_{t} = u'^{(i)}_{t} - q_t \tag{13}$$

satisfies the constant speed assumption, i.e.

$$\hat{u}^{(i)}_{t-1} = \hat{u}^{(i)}_{t} - k^{(i)}V \tag{14}$$

where $k^{(i)}$ is the *projective depth* of point *i* and it is not changed with time t. The shift q_t has been assumed to be a constant between frames. Note that q_t is taking into account the effect of $Tx/(sZ_i)$ ignored in Eq. (7), under the assumption that the depth variation in the scene is small compared to the distance of the scene from the camera, or/and the x translation component T_x is small. From Eq. (12) to Eq. (14) we have

$$\hat{m}^{(i)}_{t} = k^{(i)}V + q_t - q_{t-1} \tag{15}$$

Assume that q_t is a random variable with zero mean, then $k^{(i)}V$, which will treated as one variable, can be estimated as the mean (average) of $\hat{m}_t^{(i)}$ over all the frames in the image sequence, i.e.

$$k^{(i)}V = \frac{1}{T}\sum_{t=1}^{T}\hat{m}_t^{(i)} \triangleq \underset{t\in T}{E}[\hat{m}_t^{(i)}] \tag{16}$$

We can always assume that $q_0 = 0$, so q_t can be estimated iteratively by

$$q_t = \hat{m}_t^{(i)} - k^{(i)}V + q_{t-1} \ , \ t = 1,2,...,T \tag{17}$$

During the time period [0,T] under consideration from frame 0 to frame T, if we have N points ($i=1,2,...,N$), we will have an estimation of q_t irrelevant to any specific points, i.e.

$$q_t = \overline{m}_t - \overline{k}V + q_{t-1} \ , \ t = 1,2,...,T \tag{18}$$

where

$$\begin{aligned} \overline{m}_t &= \frac{1}{N}\sum_{i=1}^{N}\hat{m}_t^{(i)} \triangleq \underset{i}{E}[\hat{m}_t^{(i)}] \\ \overline{k}V &= \frac{1}{N}\sum_{i=1}^{N}k^{(i)}V \triangleq \underset{i}{E}\{\underset{t}{E}[\hat{m}_t^{(i)}]\} = \underset{t}{E}[\hat{m}_t] \end{aligned} \tag{19}$$

Using more than one points in Eq. (18) means a more robust estimation. After we find q_t for each frame, we just simply shift the rectified frame t by q_t, which resulting in a image sequence as if the vehicle undergoes a translation motion with constant velocity V within the time period $[0,T]$. However, in a real situation, it is rare that a point can be in the field of views of all the frames in a long image sequence. So three algorithms have been proposed in real applications.

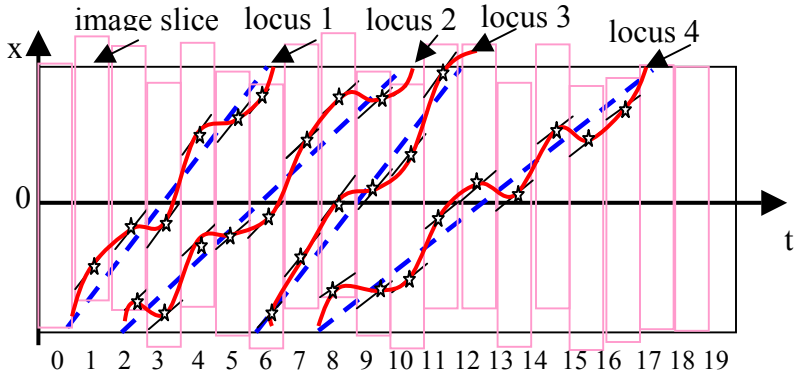


Fig. 2. Motion filtering through locus tracking and fitting

[Motion Filtering Algorithm 1 - Locus tracking and fitting approach]

The first algorithm needs to track some reliable feature points across multiple (warped) images. The basic idea of motion filtering will be presented in this algorithm. For illustrative purpose, in Fig. 2, each x-t image slice taken from frame t is presented as a light-colored rectangle with frame number labeled below it . The algorithm consists of the following steps.

Step 1. Track some reliable feature points (along the scanlines) across as many frames as possible in the rectified image sequence. The feature points can be represented as

$$\{u_t'^{(i)}, t = 0,...,T\}. i = 1,...,N$$

In Fig. 2, the feature points being tracked are marked as stars on "raw loci" represented by solid lines. Note that for a given point *i*, it only appears in a period of time $T_i \subset T$. For example, for point $i = 1$, $T_1 = \{1,2,3,4,5,6\}$.

Step 2. Compute the "displacement sequence" of each point *i* as

$$\{\hat{m}_t^{(i)} = u_t'^{(i)} - u_{t-1}'^{(i)}, t \in T_i\} \text{ (i=1,2,...,N)}$$

and find the average value $k^{(i)}V$ of *i*th displacement sequence by using Eq. (16), i.e. $k^{(i)}V = \underset{t \in T_i}{E}[\hat{m}_t^{(i)}]$. Note that $k^{(i)}V$ is the slope of the "filtered" straight locus (a dash line in Fig. 2) of point *i*, and it is computed from all the frames that the point appears. Real images of "raw locus" and "filtered locus" can be found in Fig. 5(2).

Step 3. Compute the shift q_t at each frame *t* by averaging the result from all points that appear in frame *t*, i.e.

$$\begin{aligned} q_t^{(i)} &= \hat{m}_t^{(i)} - k^{(i)}V + q_{t-1}^{(i)} \ , \ t \in T_i \\ q_t &= \underset{T_i \cap t \neq \varnothing}{E}(q_t^{(i)}) \end{aligned} \tag{20}$$

In the illustrative example in Fig. 2, q_1 is computed only by point *i*=1 (assume q_0=0), q_2 to q_5 by point 1 and 2, q_6 by point 1, 2 and 3, and so on. The computation is carried out sequentially from frame 1 to frame T since the calculation of the residual $q_t^{(i)}$ requires the value of $q_{t-1}^{(i)}$ of frame t-1. For any point *i*, the computation of $q_{t_s}^{(i)}$ in its start time t_s will use the average q_{t_s-1} in frame

t_s-1 that has been estimated before. For example, for point i= 3, t_s = 6, and $q^{(3)}_{t_s-1} = q_5$, where q_5 is estimated from points 1 and 2.

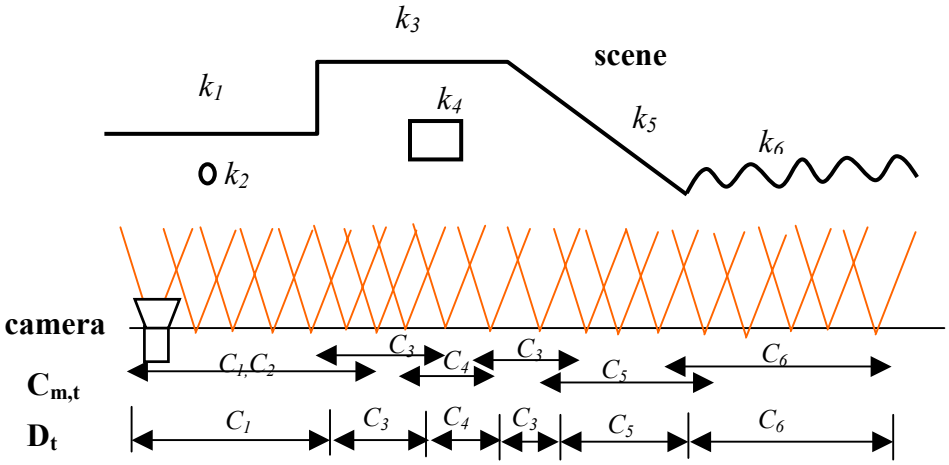


Fig. 3. Motion classification and selection

[Motion Filtering Algorithm 2 - Motion classification and selection approach]

The second algorithm is directly based on pyramid block matching algorithms and the parameter estimation results without explicitly tracking any feature points over multiple frames. The basic idea in algorithm 2 is to find in each frame t some kind of depths that will appear in the next T_t frames and that will be used to estimate the average projective depth for frame t and all the T_t frames. In frame $t \in [0,T]$, we have N_t points (i_t=1,2,...,N_t) and their displacements $\hat{m}_t^{(i_t)}$. Even if point sets may be different in different frames, we can also calculate the average displacement of N_t points in frame t:

$$\overline{m}_t = \frac{1}{N_t}\sum_{i_t=1}^{N_t}\hat{m}_t^{(i_t)} = \underset{i_t}{E}[\hat{m}_t^{(i_t)}] \tag{21}$$

and the displacement residual (the shift q_t) in each frame can be calculated

$$q_t = \overline{m}_t - \overline{k}_t V + q_{t-1} \ , \ t = 1,2,...,T \tag{22}$$

Note that in Eq. (22), the key is to find the average projective depth $\overline{k}_t$ in each frame, which is a function of t (and also of the match points in frame t) due to that each frame have different set of match points. We do not have a close-form solution if points are not tracked from frame to

frame; however, statistically, the points in successive frames do not have obvious differences in depth except in the non-overlapping bordering pixels of the two frames , thanks to the dense match points in each frame (say, one match per 16×16 block in an image). The basic idea in algorithm 2 is to find in each frame t some kind of depths that will appear in the next T_t frames and that will be used to estimate the average projective depth for frame t and all the T_t frames. The algorithm consists of the following three steps.

Step 1. *Motion classification*. For each frame, the motion vectors $\hat{m}_t^{(i_t)}$ are grouped into several classes $\{C_{m,t}\}$ according to the values of $\hat{m}_t^{(i_t)}$ (m is the index of a class).

Step 2. *Dominant motion class selection*. A dominant motion class $D_t \in \{C_{m,t}\}$ is selected for frame t, which will appear in the next T_t frames by comparing the motion classes in these frames. The average displacement and the average projective depth are calculated for frame t and all these frames as

$$\begin{aligned} \overline{m}_t &= \underset{i \in D_t}{E}[\hat{m}_t^{(i)}] \\ \overline{k}_t V &= \underset{\tau \in T_t}{E}\{\underset{i \in D_t}{E}[\hat{m}_\tau^{(i)}]\} \end{aligned} \tag{23}$$

Step 3. *Image shifting*. Calculate the displacement residuals for each frame using Eq. (22), and then apply them to the corresponding frames.

Fig. 3 illustrates the a scene with six main depths viewed by a camera. The coverage of the motion classes ($C_1 \sim C_6$) and the dominant motion for each frame t are also shown in Fig. 3.

[Motion Filtering Algorithm 3 - Statistical locus fitting / smoothing approach]

The third algorithm is also based on pyramid block matching algorithms and the parameter estimation results without explicitly tracking feature points. Again, in frame $t \in [0,T]$, we have N_t points ($i_t=1,2,\ldots,N_t$) and their displacements $\hat{m}_t^{(i_t)}$. We want to directly use Eq. (21) and Eq. (22) to estimate the displacement residual in each frame. With some approximation, we can use the following rather simple filtering methods:

(1). *Median filter* - In the time period $[0,T]$ of the image sequence, the average of the 60% median $\hat{m}_t^{(i_t)}$ is calculated as the average displacement $\overline{m}_t$ in frame t by using Eq. (21), and piece-wise straight lines are fitted to displacement sequence $\{(\overline{m}_t, t)\}$ in order to estimate $\overline{k}_t V$.

(2). *Gaussian Low-pass filtering*. In this method, $\bar{k}_t \mathrm{V}$ is estimated as the Gaussian low-pass filtering part of the average displacement $\overline{m}_t$ in Eq. (21). It is based on the observation the average projective depth should form a smooth curve even in the case of very complex depth variation due to the constant speed motion.

The nice thing about our automatic image stabilization is that we only need to fit a global model (Eq. (4)) to the motion fields, and only several reliable image matches (say, more than 6 pairs) between two frames are needed to stabilize a image sequence. For the current implementation, we use a pyramid-based correlation match method to find the motion vector of each representative block, which is selected by its texture measurement. Robust estimation method is used to reduce the negative effect of outliers by using both the correlation measurements and texture measurements as weights. We have found by many experiments of real image sequences that Algorithm 3 is the simplest and most robust motion filtering algorithm among the three and is good enough for generating the required stabilization results. Further work is need to apply Algorithm 1 and Algorithm 2 to more real image sequences.

2.2.4. PVI and EPI Generation: Examples

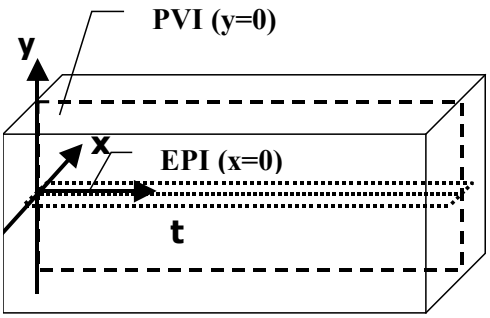
Without loss of generality, the effective focal length f of the camera is assumed to be fixed for a stabilized image sequence and both s_x and s_y are assumed to be equal to $1/f$, so Eq. (2) becomes $(x, y) = (u, v)$. From now on, we will use (x, y) instead of (u, v). Thus the stabilized image sequence obeys the following spatio-temporal (ST) perspective projection model

$$x(t) = f\frac{X + Vt}{Z}, y(t) = f\frac{Y}{Z} \tag{24}$$

where (X,Y,Z) represent the 3D coordinate at time t=0. A feature point (x,y) forms a straight locus and its depth is

$$D = Z = f\frac{V}{v} = f\frac{Vdt}{dx} \tag{25}$$

where $v = dx/dt$ is the slope of the straight locus.



(1) ST image model

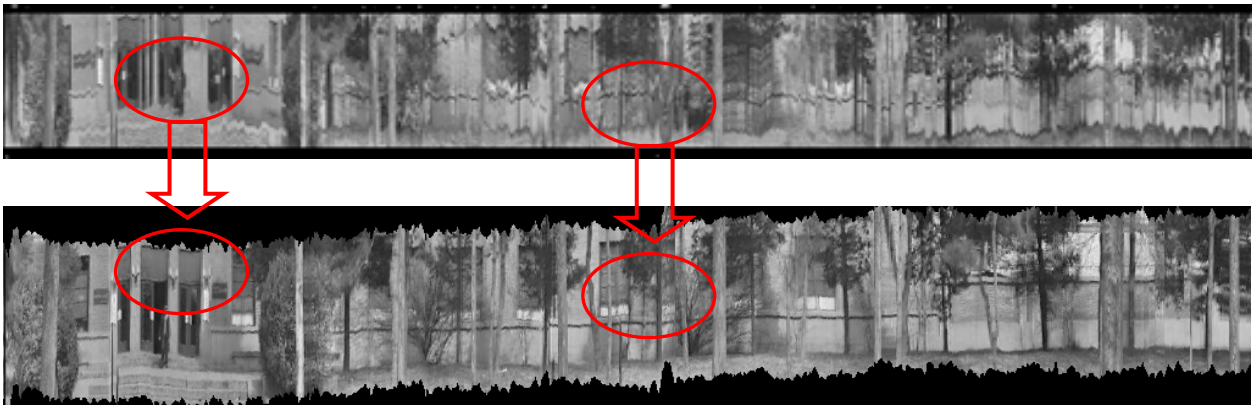
(2) Stereo PVIs (x = 0 and x= -56)

(3). Real ST images

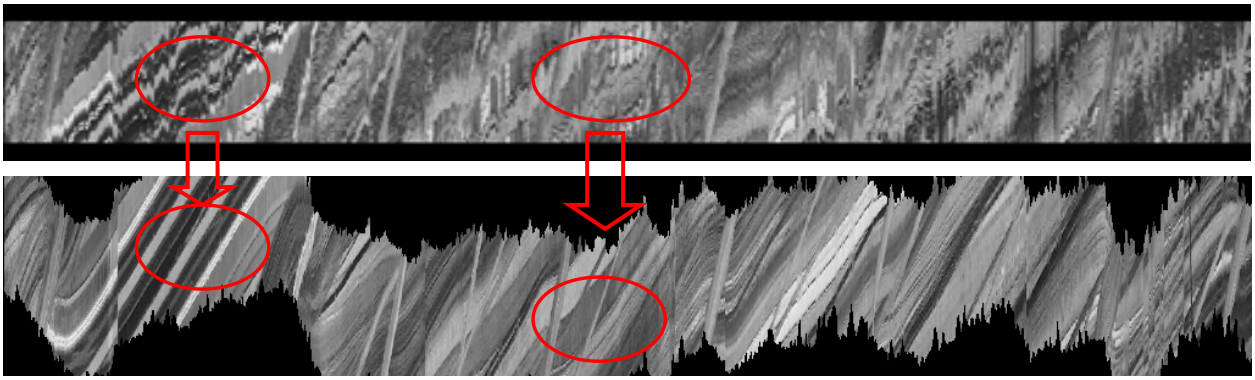
Fig. 4. ST images from an image sequence

In order words, after the image stabilization, two kinds of useful 2D ST images can be extracted (Fig. 4). One is the Panoramic View Image (PVI), which possesses most of the 2D information of a roadside scene. The other is the Epipolar Plane Image (EPI), whose ST texture orientations represent depths of scene points. Fig 4(2) shows two PVIs that are extracted from x=0 and x=-56. They are parallel-perspective images with multiple viewpoints in the t axis, and depth information can be derived from this ST stereo pair. However, 3D recovery of ST stereo faces the same problems as in traditional stereo: in addition to the correspondence problem, occluding region can not be easily handled. Thus it is important to make use of the continuos information

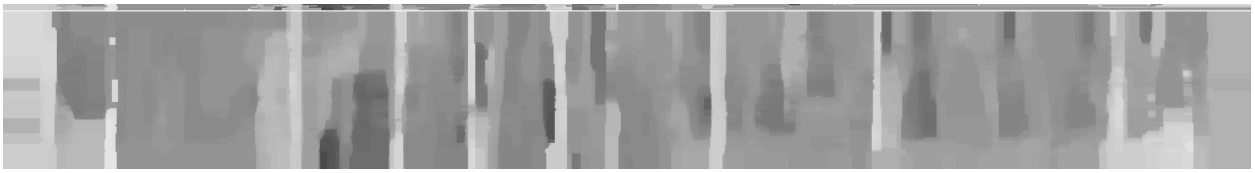
in the epipolar plane images. Fig 4(3) shows an EPI (x = 9) from which dense depth map can be reconstructed by the proposed panoramic EPI analysis method.



(1) PVI (x=0) without and with stabilization



(2) EPI (y = 0) without and with stabilization

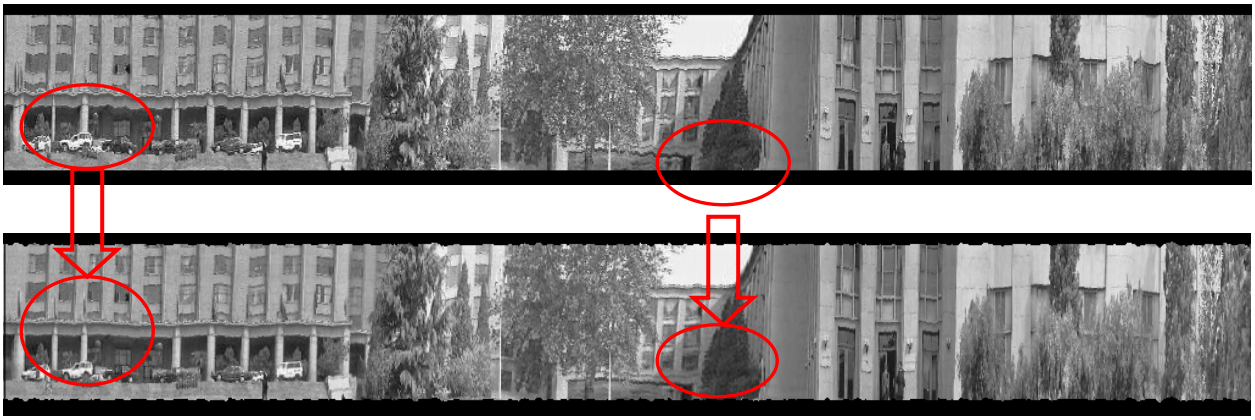


(3) Panoramic depth map (the nearer, the brighter)

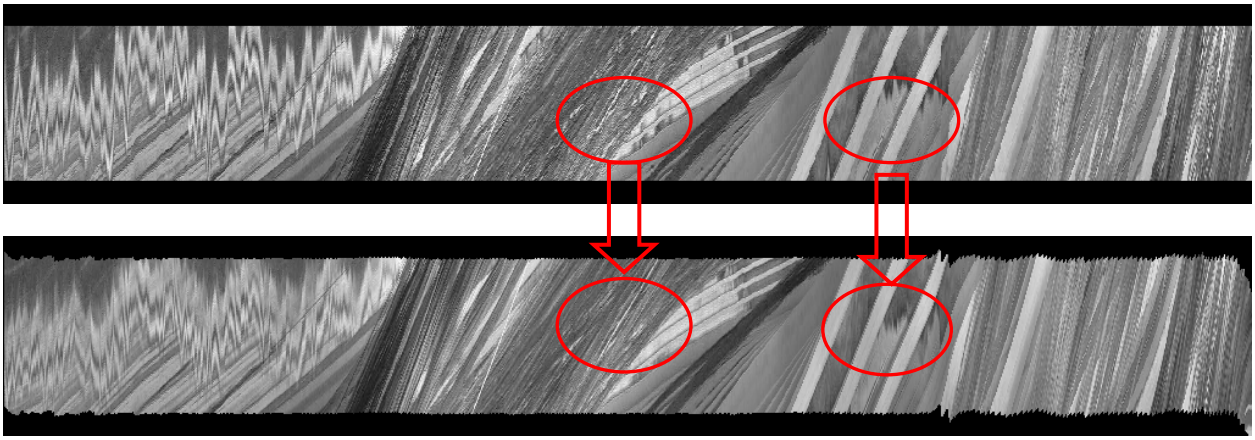
Fig.5. Stabilization of the TREE sequence (128*128*1024)

Fig. 5 shows PVIs and EPIs without and with image stabilization for a TREE sequence, which consists of 1024 frames of 128×128 images. It is obvious that stabilization plays a vital role in the construction of good panoramic and epipolar plane images when the camera's fluctuations are large as in this example. In this experiment, median motion filtering method is used. The depth map (Fig. 5(3)) can be obtained through our epipolar plane image analysis on the stabilized EPIs (Section 3), which is almost impossible without image stabilization (see Fig. 5(2)). Fig. 6 shows the stabilization results of a BUILDING sequence when tiny fluctuations

occurred. Better PVIs and EPIs are obtained after image stabilization by using a Gaussian low-pass filtering with a 200-frame Gaussian window. Fine stabilization results can be seen, for example, in the elliptic marked regions. The waterfall-like texture patterns in the left of the EPI in Fig. 6(2) are due to the aperture problem – this EPI corresponds to the position along a horizontal ridge of the building. A video-clip of an additional image sequence can be found at http://www.cs.umass.edu/~zhu/btvstabi.mpg to demonstrate the difference between original and stabilized ones dynamically.



(1) PVI (x=24) without and with stabilization



(2) EPI (y = 0) without and with stabilization

Fig. 6. Stabilization of the BUILDING sequence

2.3. Panoramic EPI Analysis Approach

Spatio-temporal panoramic view images(PVIs) provide a compact representation for large-scale scene. Stereo PVIs can be used to estimate the depth information of the scene. The difference between panoramic stereo and the traditional stereo is that panoramic images are

parallel-perspective projections. The depth of a point is proportional to the "displacement" in the t direction in a pair of stereo PVIs (Fig. 4 (2); in Eq. (25) "disparity" dx is fixed and $D \propto dt$), which means that depth resolutions are the same for different depths (Zhu et al., 1999a). However, there are some disadvantages when we use stereo PVIs to recover the depth of a scene. First, stereo PVI approach faces the same correspondence problem as in any traditional stereo methods. Second, occluding regions in two panoramic views cannot be easily handled due to the lack of information. The solution to these two problems is to effectively use the information in between, i.e. that of the epipolar plane images. Our panoramic epipolar plane analysis method consists of three important parts: orientation detection, motion boundary localization and occlusion recovery. It has the following three advantages. (1) It is roust applying to a complex natural scene and a motion with fluctuation. With a spatio-temporal-frequency domain analysis, no feature detection, hard thresholding and locus tracking are used in our algorithm. (2) It is efficient in that it only processes a small fraction of the necessary data instead of the entire 3D ST images. (3). It is versatile since occlusion and textureless regions have been handled. In this section, after we derive the motion occlusion model in spatio-temporal and frequent domain, we will describe each module in details.

2.3.1 Motion Texture and Motion Occlusion Models

The 1st order motion texture model of an EPI can be expressed in the spatio-temporal domain as (Allmen and Dyer, 1991; Adelson and Bergen, 1985; Heeger, 1987)

$$g(x,t) = f(x - vt) \tag{26}$$

where $f(x)$ is the image of a single scan line at time t =0. By Fourier transform, the model in the frequency domain can be derived as

$$G(\xi, \omega) = F(\xi)\, \delta(v\xi + \omega) \tag{27}$$

which states that object points with the same depth values and the same constant translation occupy a single straight line passing through the origin in the frequency domain, i.e., $v\xi + \omega = 0$. It is well-known that orientation can be easy to detect in the frequency domain than in the spatio-temporal domain when a single orientation is presented in the window of the processing [Jahne, 1991]. In this paper we will deal with multiple orientations due to depth changes. We model the motion occlusion in an x-t image (EPI) as

$$g(x,t) = m_s(x,t)g_1(x,t) + (1 - m_s(x,t))g_2(x,t) \tag{28}$$

where the first layer $g_1(x,t)$ is occluded by the second layer $g_2(x,t)$, and $m_s(x,t)$ is a occluding mask (Fig. 7). Under a 1st order translation, the ith layer with velocity v_i can be expressed as

$$g_i(x,t) = f_i(x - v_i t), (i = 1,2)$$

where $v_1 < v_2$. The occluding mask is a step function moving with velocity v_2, i.e.,

$$m_s(x,t) = u(x - v_2 t) \tag{29}$$

and the value of the function is 0 or 1. Hence the *1st order motion occlusion model* can be written as

$$g(x,t) = u(x - v_2 t) f_1(x - v_1 t) + (1 - u(x - v_2 t)) f_2(x - v_2 t) \tag{30}$$

and the Fourier transform of the model can be derived as (Appendix 2.3)

$$G(\xi,\omega) = \frac{1}{v_1 - v_2} F_1(\frac{v_2\xi + \omega}{v_2 - v_1}) U(\frac{v_1\xi + \omega}{v_1 - v_2}) + F_{u2}(\xi)\delta(v_2\xi + \omega) \tag{31}$$

where $F_{u2}(\xi) = F_2(\xi) - F_2(\xi) * U(\xi)$ is the Fourier transform of $f(x)(1-u(x))$, the visible parts of $f(x)$, and $U(\xi)$ is the Fourier transform of $u(x)$. Without loss of generality, we assume $u(x) = \begin{cases} 0, x < 0 \\ 1, x \geq 0 \end{cases}$, so we have

$$U(\xi) = \frac{1}{j\xi} + \pi\delta(\xi) \tag{32}$$

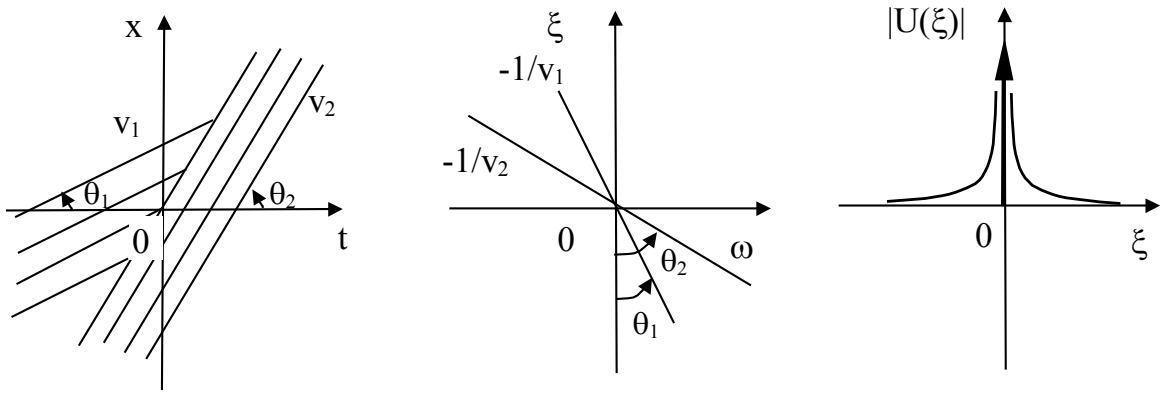
which implies that the peak value of $U(\xi)$ appears at $\xi = 0$ (Fig. 7(3)). From Eq. (31) and (32) we can conclude that most of the energy spectra lie in line $\xi = -\omega / v_1$ and line $\xi = -\omega / v_2$, which correspond to the two depth layers(Fig. 7(2)) . Fourier transforms along these two lines are

$$G(\xi, -v_1\xi) = \frac{1}{v_1 - v_2} F_1(\xi) U(0)$$

which displays a peak corresponding to the occluding layer and

$$G(\xi, -v_2\xi) = F_{u2}(\xi) + \frac{1}{v_1 - v_2} F_1(0) U(\xi)$$

which shows a peak corresponding to the occluded layer, with an addition that only has obvious effect when $\xi = 0$.



(1) *g(x,t)* (2) $|G(\xi,\omega)|$ (3) $|U(\xi)|$

Fig. 7. The motion occlusion model

2.3.2 GFOD: Gaussian-Windowed Fourier Orientation Detector

In order to detect multiple orientations more precisely and robustly, a Fourier transform is performed in a large ST window on an EPI. Simple calculation shows that the angle resolution is about 1° if the window size $m \times m$ is 64×64. Obviously, however, all the oriented textures in the large window will contribute to the energy spectrum. So for a multiple orientation pattern, multiple peaks could be detected when the window slides thorough a rather wide region near the depth boundary. For example, in Fig. 8 (1), two peaks can be detected from frame 296 to frame 322. Therefore, a Gaussian-Fourier Orientation Detector (GFOD) is designed in order to keep the precision for both orientations of motion textures and localization of motion boundaries. If the spatio-temporal Gaussian window is defined as

$$w(x,t) = \exp(\frac{x^2+t^2}{2\sigma^2})$$

where $\sigma^2 = \frac{m-1}{4}$, then the windowed motion texture can be represented as

$$g(x,t) = f(x - vt)w(x,t) \tag{33}$$

and its Fourier transform can be derived in the same way as for Eq. (31) :

$$G(\xi,\omega) = c(v)F_w(\frac{\xi - v\omega}{v^2+1})W_v(v\xi + \omega) \tag{34}$$

where $c(v) = \frac{2v}{v^2+1}$,

$$F_w(\omega) \Leftrightarrow f_w(x) = f(x)e^{-\frac{x^2}{2(\sigma\sqrt{v^2+1})^2}}$$

$$W_v(\omega) = e^{-2(\sigma/\sqrt{v^2+1})^2\omega^2} \Leftrightarrow w_v(t) = e^{-\frac{t^2}{2(\sigma/\sqrt{v^2+1})^2}}.$$

Again, most of the energy lies in the line $\xi = -\omega / v$, where

$$G(\xi,-v\xi) = F_w(\xi)W_v(0)$$

since $W_v(\omega)$ is a Gaussian function with peak at $\omega = 0$.

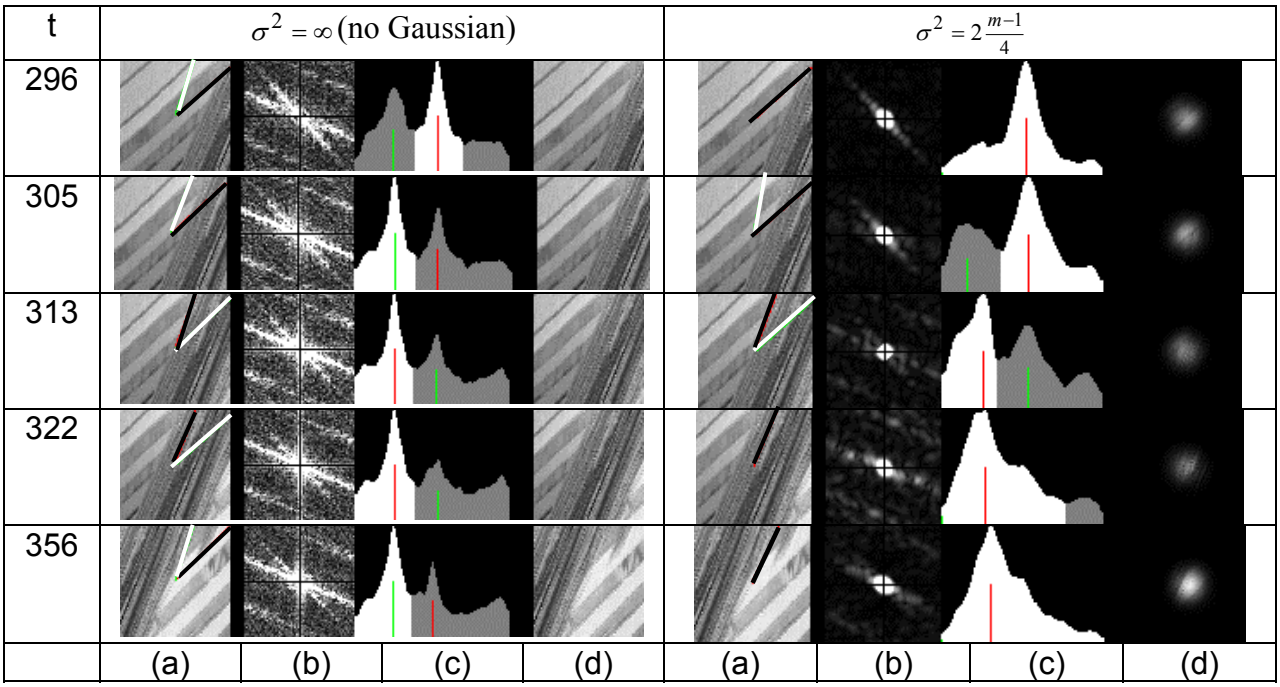
The derivation of the energy model for cases of multiple orientation and motion occlusions is quite complex; however, similar conclusion can be drawn. Here we only give a qualitative analysis. From the principle of the Fourier transform, the multiplication of a Gaussian window *w(x,t)* in the ST domain is equivalent to the convolution of a Gaussian function W(ξ,ω) in the frequency domain, which will smooth the energy distribution. The larger the covariance, the more smoothing to the energy spectrum[3]. By applying the Gaussian window, the ST patterns that are farther from the center of the window has less contribution to the final energy spectrum, but they are not eliminated. So the design of the GFOD operator tries to reach a balance between the orientation resolution (over a large window) and the localization accuracy of depth boundaries (in the center of the window).

Representing the "energy spectrum" $P(\xi,\omega)$=log$(1+G^2(\xi,\omega)$ in the polar coordinate system (r,ϕ) by a coordinate transformation $r = \sqrt{\xi^2 + \omega^2}, \phi = \frac{\pi}{2} + \arctan\left(\frac{\xi}{\omega}\right)$, we obtain a polar representation $\mathrm{P}(r,\phi)$ and an orientation histogram

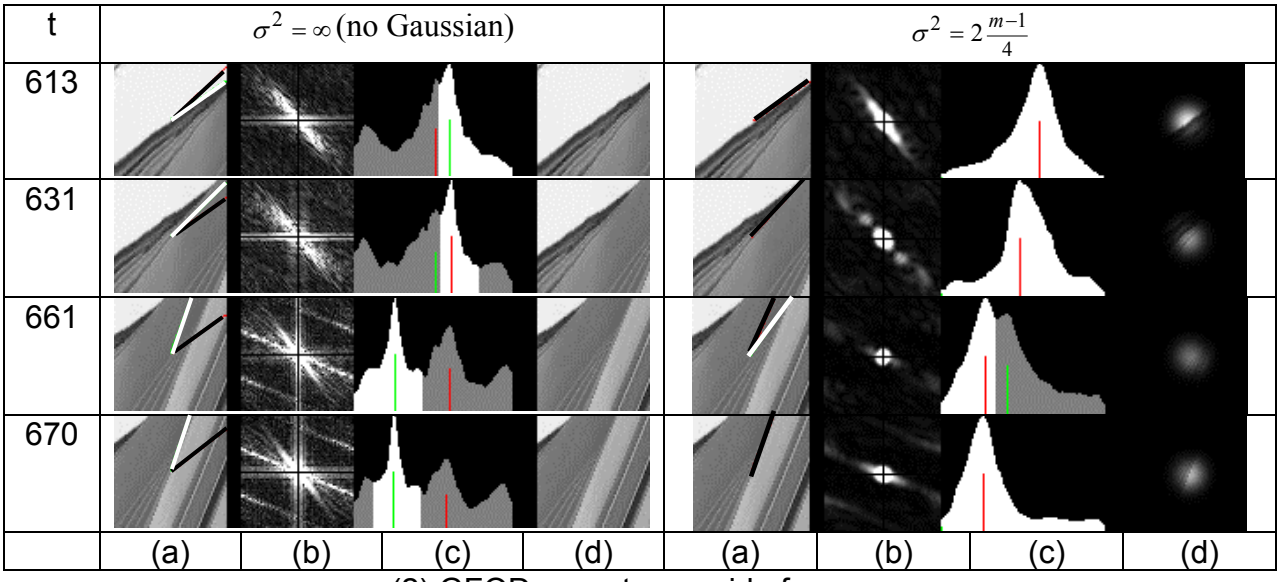
$$P_d(\phi) = \int_{r_1}^{r_2} P(r,\phi)dr\phi \in [0,\pi] \qquad (35)$$

where ϕ corresponds to the orientation angle of a ST texture and $[r_1,r_2]$ is a frequency range of the bandpass filter, which is selected adaptively according to the spatial-temporal resolution of the image. Initially, r_1 and r_2 are set to 8 and 30 respectively for a 64x64 window. An orientation energy distribution map $P_d(\phi,t)$ can be constructed, which visually represents the depths of the points along a scan line corresponding to the processed EPI.

Fig. 8 and Fig. 9 compare experimental results for the BUILDING sequence using a rectangular window ($\sigma^2 = \infty$) and a Gaussian window ($\sigma^2 = 2\frac{m-1}{4}$). For example, in Fig. 8 (1), two peaks can be detected within a large neighborhood (27 frames from frame 296 to frame 322) of a depth boundary at frame 131 when a rectangle window is used. By using the Gaussian window, the Fourier spectrum is smoothed; however motion boundaries can be located in a much smaller range. In Fig. 8(1), two peaks are detected only in 8-frame intervals from frame 305 to frame 313 without degrading the angular resolution of orientation. This is because the ST texture off the center is decreased but is not eliminated by using a Gaussian window. Note that multiple orientations are still detected both at and near motion boundaries even if Gaussian windows are used. Therefore, motion boundaries are further localized by using a spatial orientation selection method presented in Section 3.4, which results in the final dense histogram of the orientation angles in Fig. 9. Fig. 8(2) shows that the detecting errors by the rectangle windowed Fourier method could not even be correctly selected by using the motion boundary localization method in Section 3.4. This set of images is taken from the EPI of a region with gradually changing depths (side façade of a building - see Fig. 6 and Fig. 9). The reason for this kind error is that the stronger oriented texture off the center has different orientations from the weaker motion texture at the center. In frame 661, none of the two orientations detected is that of the locus in the center; but rather they are the orientations of the loci of the left and right stronger textures. In frames 613, 661 and 670, the correct orientation is among the two detected peaks, but the orientation selection by the algorithm in Section 3.4 is not correct since it is almost textureless along one of the orientations (refer to Section 3.4 for how to select an orientation). However, if the Gaussian windowed Fourier orientation detector is used, the correct orientations can be found in all the three frames (613, 661 and 670); the weak but correction orientation is detected by the GFOD in frame 661, even though the final selection in this case need further study.

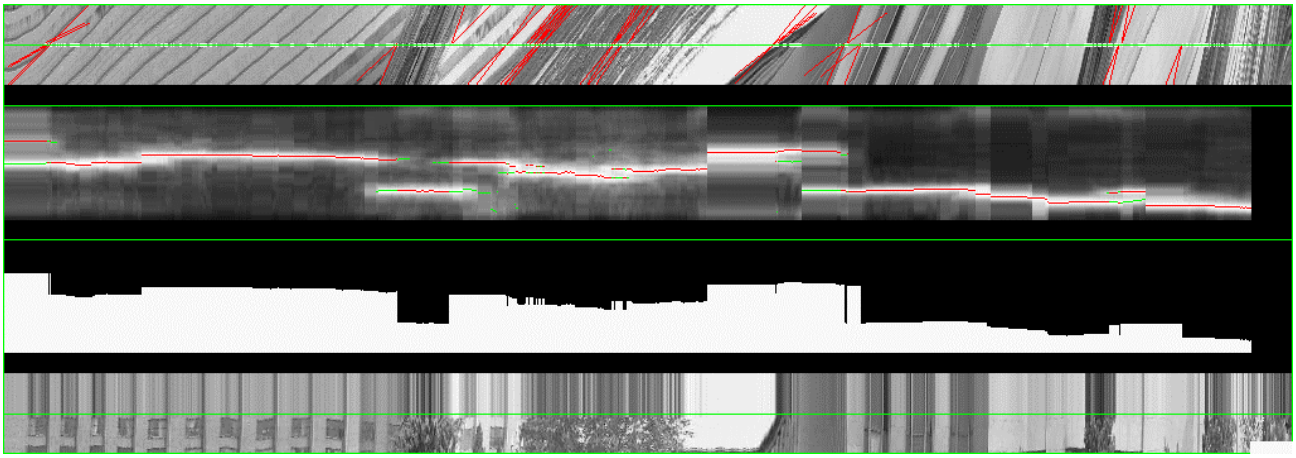


(1) GFOD operator on occluding boundary

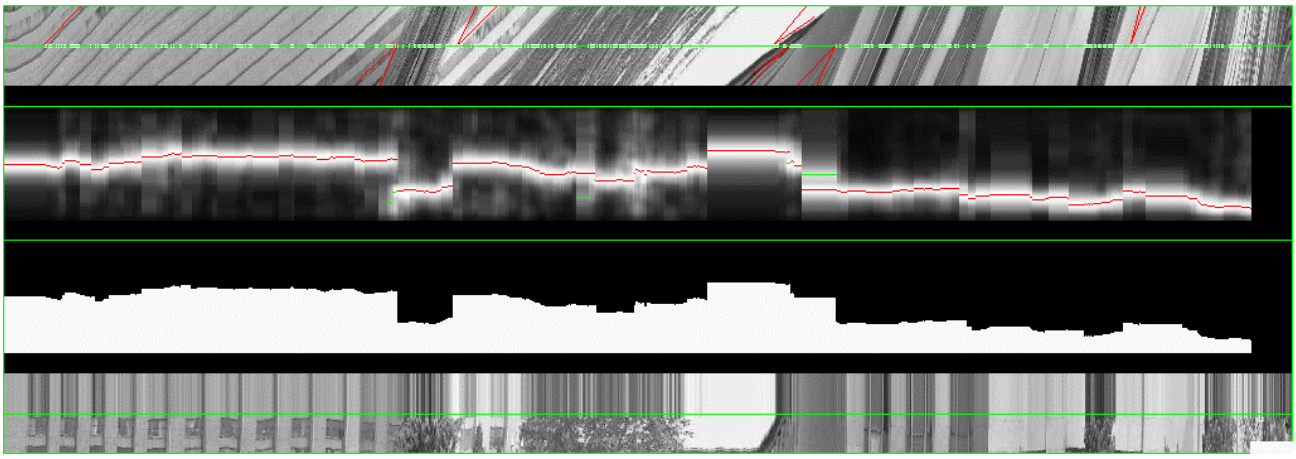


(2) GFOD operator on side face

Fig. 8. Multiple orientation Detection using GFOD. The frame index (t) corresponds to the time in the EPI shown in Fig. 9. In both (1) and (2): (a) 64×64 x-t image block superimposed by multiple orientation vectors (the darker line is the final orientation for the point in the center selected by the spatial comparison method in Section 3.4). (b) energy spectra (c) orientation histogram with the detected peak(s). (d) Gaussian weighted x-t image.



(1). Orientation detection using a rectangular window



(2) Orientation detection using a Gaussian window

Fig. 9. Multiple orientation detection by GFOD (rows 1-4 of both (1) and (2): x-t image (EPI, y = 56), orientation energy distribution map (the darker one as the selected peak, lighter one as the second peak), histogram of orientation angles and part of the corresponding PVI (PVI x = 24 is selected in order to shown the side face of building around frame 613; the horizontal line in the PVI corresponds to the EPI in the first row)

2.3.3. Depth Belief Map and Data Selection

Basically, our panoramic epipolar plane analysis method processes only the EPI data around a panoramic view image (e.g., the centered horizontal line in the EPI of Fig. 9). Small amount of additional data is processed only for the motion boundaries (see Fig. 12 in Section 3.5). Moreover, the algorithms also deal with the following two problems: the aperture problems of horizontal edges that run along the motion direction, and depth interpolation in textureless

regions. Depth estimations at vertical edge points are more robust. To take this observation into account, a belief map corresponding to a PVI $I_{PVI}(y,t)$ is calculated as

$$B(y,t) = \frac{\partial I_{PVI}(y,t)}{\partial t} - \frac{\partial I_{PVI}(y,t)}{\partial y} \tag{36}$$

(1). Panoramic intensity image (x =0)

(2). Panoramic belief map

Fig. 10. Panoramic belief map

Fig. 10(2) shows the belief map corresponding to the PVI in Fig. 10 (1). The brighter intensity in the belief map shows stronger belief. The basic data selection is as follows. For the epipolar plane image $I_{EPI}(x,t)$ corresponding to a y coordinate of a given PVI, orientations are detected only at the x coordinate from which the panorama has been taken (typically $x = 0$). The GFOD is applied only to each location (x,t_i) where the belief value $B(y,t_i)$ is greater than a given threshold; typically it is a very small value (e.g., 2). A fast GFOD algorithm of such that use the temporal overlapping of the successive Gaussian windows is given in Appendix 2.4. Single or multiple orientation angles $\theta_k (k = 1,\cdots,K)$ are determined by detecting peaks in an orientation histogram. Image velocity can be calculated for each orientation as $v_k = \tan\theta_k$. A motion boundary will appear within the Gaussian window if the orientation number K is greater than 1 (K=2 for double peaks). The additional data selection in motion boundary localization, depth interpolation, occlusion recovery and resolution enhancement will be given in the next two subsections.

2.3.4. Motion Boundary Localization and Depth Interpolation

Since multiple orientations are detected not only at but also near the motion boundaries by using the large GFOD operator, a *Motion Boundary Localizer* is designed to verify if the motion boundary is right in the center of the Gaussian window. In order that the method is valid for most of the cases encountered in a natural scene and applicable to the EPIs generated by a fluctuating camera, we use a different approach other than locus tracking which often fails due to the non-ideal ST textures from a complex scene and a real motion. In our algorithm, multiple scale intensity similarities are measured along the detected orientations $\theta_k (k = 1, \cdots, K)$, and the orientation with the greatest similarity measurement is selected as the right orientation. Note that only a "comparison and selection" operation is used without assuming any feature points detecting and thresholding.

Consider the case in which two orientations θ_1 and θ_2 ($\theta_1 > \theta_2$) are detected within a Gaussian window. Dissimilarity measurements along θ_1 and θ_2 for a given circular window of radius R centered at the point (x_0,t_0) are defined as (Fig. 11 (a) and (b); refer to Fig. 8)

$$C_{\pm}(\theta_k, R) = \frac{1}{R}\sum_{r=1}^{R} I^2(\pm r, \theta_k) - \bar{I}_{\pm}^2(\theta_k, R) \quad \text{(k=1,2)} \tag{37}$$

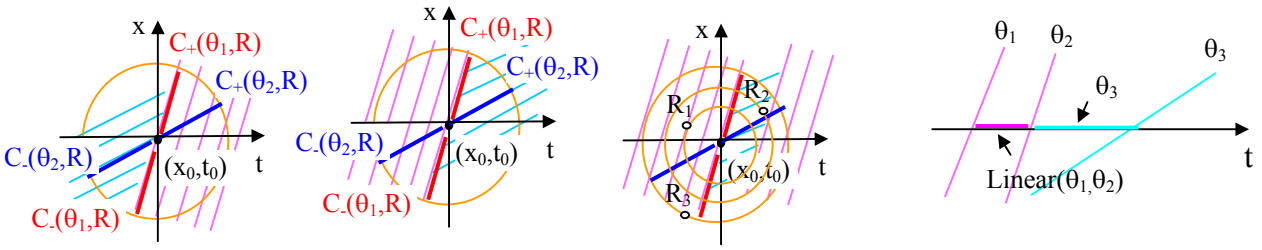
where $r = \sqrt{(x - x_0)^2 + (t - t_0)^2}$, $\bar{I}_{\pm}(\theta_k, R) = \frac{1}{R}\sum_{r=1}^{R} I(\pm r, \theta_k)$. Sub-indices '+' and '-' denote the dissimilarity measurements along the detected orientations in positive and negative x directions respectively. This is designed for dealing with occlusion of a far object (θ_2) by a near one (θ_1). The dissimilarity measurements for near and far objects are defined as

$$\begin{aligned} E(\theta_1, R) &= \frac{1}{2}\left(C_+(\theta_1, R) + C_-(\theta_1, R)\right) / P_d(\theta_1) \\ E(\theta_2, R) &= \min(C_+(\theta_2, R), C_-(\theta_2, R)) / P_d(\theta_2) \end{aligned} \tag{38}$$

respectively, where weight $P_d(\theta_k)$ is the value of the orientation histogram (Eq. (35)) at θ_k (k=1,2). The higher the value is, the lower the dissimilarity measurement will be. A verification criterion can be expressed as

$$\theta = \begin{cases} \theta_1, & \text{if } E(\theta_1, R) \le E(\theta_2, R) \\ \theta_2, & \text{Otherwise} \end{cases} \tag{39}$$

The condition of occlusion and reappearance can be judged either by comparing $C_+(\theta_2, R)$ and $C_-(\theta_2, R)$ (see Fig. 11), or by analyzing the context of the processing (i.e., the change of depths - refer to Fig. 12). In case of occlusion of a far object by a near object (far to near, Fig. 11(a)), we have $C_-(\theta_2, R) < C_+(\theta_2, R)$, and in reappearance (near to far, Fig. 11(b)), we have $C_+(\theta_2, R) < C_-(\theta_2, R)$.



(a). occlusion (b) reappearance (c) multi-scale window (d) depth interpolation

Fig. 11. Principle of the depth boundary localization and depth interpolation

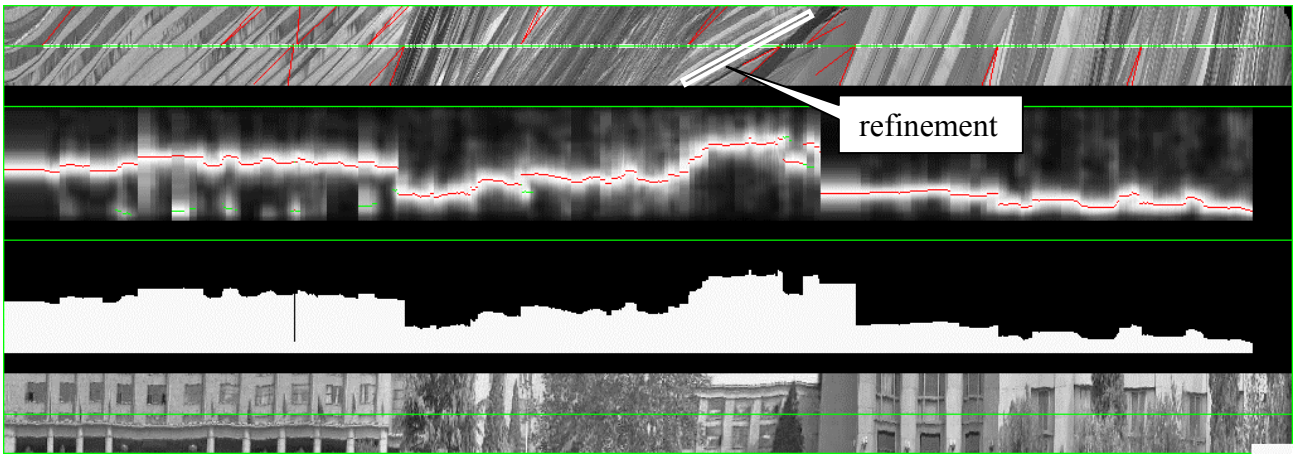


Fig. 12. Multiple orientation detection, depth boundary localization and depth interpolation (rows 1-4: x-t image (EPI, y = 9) with the processed points and depth boundaries, orientation energy distribution map, histogram of orientation angles and part of the corresponding PVI (x = 0; the horizontal line is corresponding to the EPI in the first row)

In order to handle cases of different object sizes, motion velocities and object occlusions, multiple scale dissimilarity measurements $E(\theta_k, R_i)$ (e.g., i=1,2,3) are calculated within multiple scale windows of radii R_i (i=1,2,3), $R_1 < R_2 < R_3$. In our experiments, we have selected

$R_1=m/8$, $R_2=m/4$, $R_3=m/2$ ($m = 64$ is the window size; see Fig. 11(c)). By defining the following ratio

$$D_i = \frac{\max\left(E(\theta_1, R_i), E(\theta_2, R_i)\right)}{\min\left(E(\theta_1, R_i), E(\theta_2, R_i)\right)} \tag{40}$$

scale p (p=1,2,3) with maximum D_p is selected for comparing the intensity similarities. For example, in Fig. 11(c), R_2 will be selected.

The selected orientation angle θ can be refined by searching for a minimum dissimilarity measurement for a small-angle range around θ. The accuracy of the orientation angle, especially that of a far object, can be improved by using more possible frames. The frame number can be decided by examining the occluding relations near the far object (e.g., the left part of the EPI in Fig. 9 and the indicated location in Fig. 12). In order to obtain a dense depth map, interpolations are applied to textureless or weak-textured regions /points where no orientation can be detected). The proposed interpolation method (Fig. 11(d)) is based on the fact that depth discontinuity almost always implies an occluding boundary or shading boundary. The value $\theta(t)$ between two instants of time t_1 and t_2 with estimated orientation angles θ_1 and θ_2 is linearly interpolated for smooth depth change (i.e., $|\theta_1 - \theta_2| < T_{dis}$, T_{dis} is a threshold), and is selected as $\min(\theta_1, \theta_2)$, i.e. , the angle of the farther object, for depth discontinuity (i.e., $|\theta_1 - \theta_2| \geq T_{dis}$). The processing results of a real x-t image(EPI) are shown in Fig. 9 and Fig. 12 by histograms of orientation angles.

2.4. **Panoramic Modeling and Generalized Landmark Selection**

Our 3D panoramic scene modeling approach consists of four modules: (1) Image rectification and stabilization, (2) depth map acquisition, (3) fusion of the depth and intensity maps, and (4) landmark selection. The system diagram is shown in Fig. 13. We will discuss each module with the results of real image sequences.

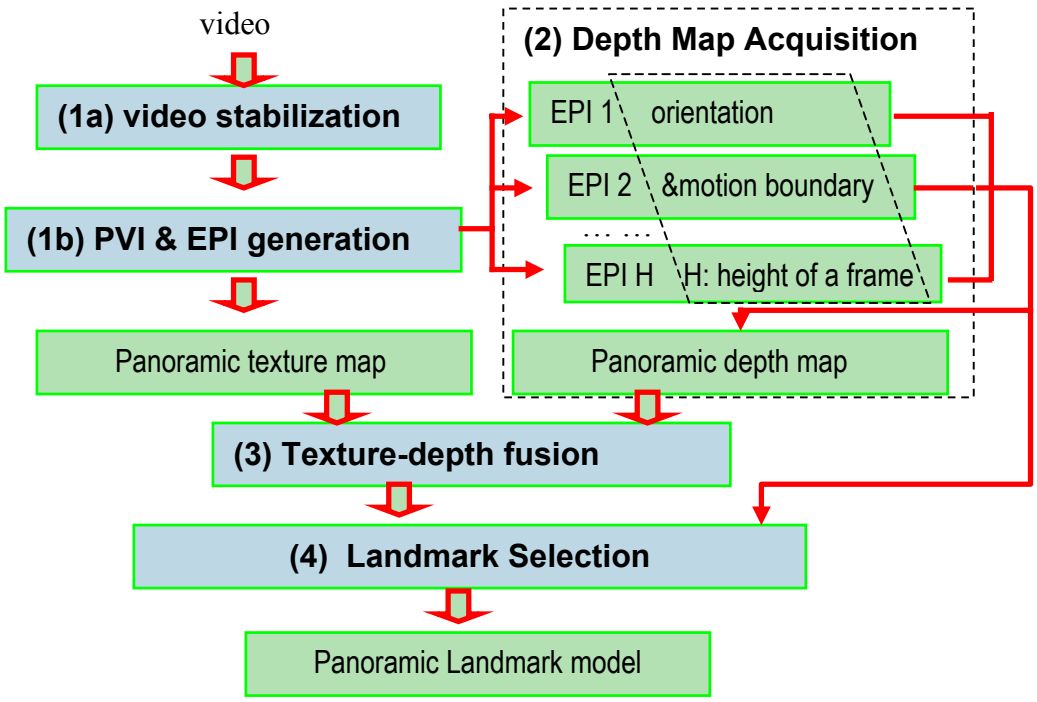
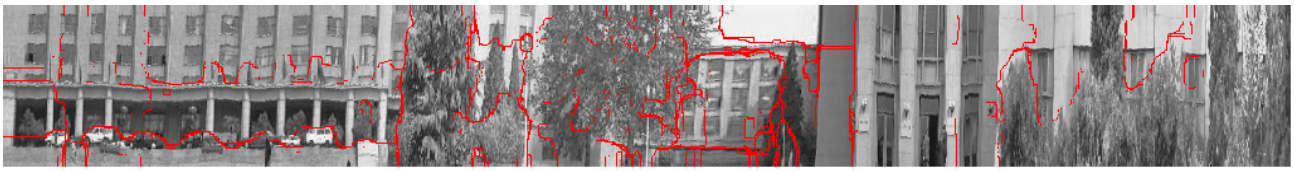


Fig. 13. System diagram

2.4.1. Image Rectification and Stabilization

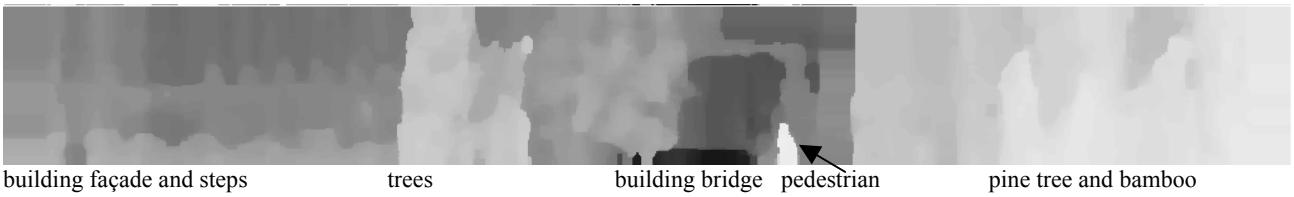
In order to compute absolute depths of objects in a scene, it is necessary to calibrate the camera. Fortunately, accurate intrinsic and extrinsic parameters of the camera are not a necessity for our purpose and our method. We assume that the optical axis passes through the center of the image and the approximate focal length f can be easily determined by a simple calibration procedure. The accuracy of f is not so important. In our approach, it is only used twice. First it is used to approximately decomposition the translational components (motion parallax) and the rotation angles (in Eq.(7)). In our experiments, we can assume the fixed focal length (i.e., s =1) and 1:1 aspect ratio (i.e., $s_x=s_y$ = 1/f). Note that the motion filter step after the first image rectification step using such decomposition will compensate the errors introduced by inaccurate focal lengths and other approximations. Second, it is used to compute the depth in Eq.(25), which is only a scale factor for depths of all the points. Neither do we need to acquire the extrinsic parameters of the camera explicitly. What we need is to rectify each image as if the horizontal X axis of the camera is along the direction of motion. This can be easily done by a pure image rotation transformation by referencing an original image in an image sequence to a known rectangular planar surface in the scene whose horizontal edges are parallel to the motion direction, instead of actually measuring the coordinates of any 3D objects (Zhu, 1997). Such surface patch may be a window on a building's facade. Now we have a “software-adjusted”

camera whose motion is satisfied the motion model in Section 2.1. The image stabilization operation can then be applied to the rectified image sequence. The stabilization method and experimental results have been given in Section 2, and image stabilization have been implemented at 30 Hz for 256×256 images (Zhu et al., 1998b).

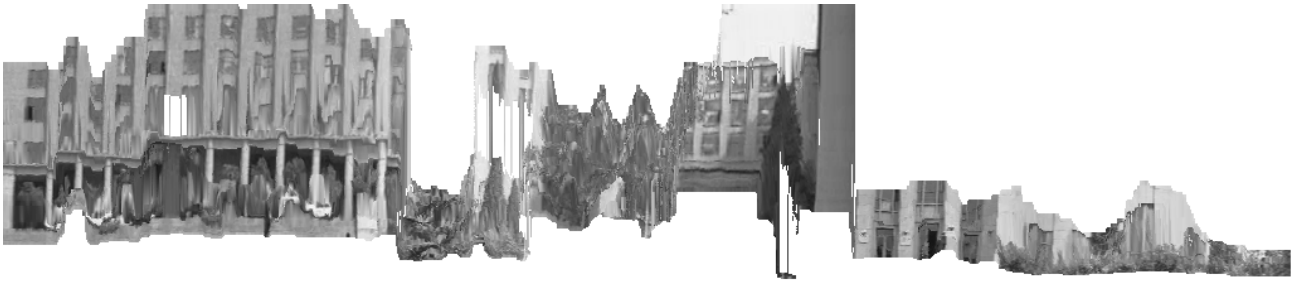


(1) depth boundaries (red lines) overlay on the panorama

(2) original panoramic depth map



(3) panoramic depth map after depth-intensity fusion



(4) parallel projection of the 3D panorama

Fig. 14. Panoramic depth map for the BUILDING sequence

2.4.2. Panoramic Depth Map Acquisition

Suppose that a video sequence has F frames, each of size $W \times H$, and the size of the Gaussian window is $m \times m$. The panoramic depth map corresponds to a panoramic view image (PVI). The H×F depth map is acquired by the independent and parallel processing of H images of 2D panoramic epipolar planes (Fig. 13). After the belief map for depth measurement (e.g., Fig. 10) is calculated from the panorama, depth information for each scan line of the PVI is obtained by executing the algorithms of multiple orientation detection, motion boundary localization and depth interpolation in the corresponding epipolar planes, as described in Section 3. Fig. 14(2) shows the original panoramic depth map of the BUILDING sequence. The nearer depths are represented by brighter intensities.

2.4.3. Fusion of Depth and Intensity map

It has been pointed out that depth information cannot be completely recovered using only the motion cue (Black and Jepson, 1998). A deeper understanding of the fusion of motion and texture needs further study. In this paper, a simple two-step algorithm was used:

(1). Median filtering on the depth map preserves each depth boundary while eliminating errors due to aperture problems and complex non-rigid motion of trees, etc.

(2). Intensity boundaries and depth boundaries are labeled in vertical directions. If there is no intensity boundary at a depth boundary, then the depth boundary is moved to the location of a most suitable intensity boundary.

Fig. 14(3) shows the fusion result for the BUILDING sequence. Depth boundaries of the depth map, superimposed on the panoramic intensity image as red lines in Fig. 14(1), shows the accuracy of localization. A parallel projection of the 3D panorama in Fig. 14(4) visualizes the depth estimates. The panoramic depth map of the more complex TREE sequence has been shown in Fig. 5(3) where three distinctive depth layers of the trees can be observed.

2.4.4. Generalized Landmark Selection

There are two important issues in landmark localization for a mobile robot: where to look at and what to look at? The first is the viewing direction issue, and the second is the landmark selection issue.

Let us consider the viewing direction issue first. In the camera model of Section 2.1, we assume that the camera points perpendicular to the road side. In that case, the scene model based

on panoramic view images will be the same for the driving in both directions if we do not consider other dynamic objects (e.g. vehicles, people) on the roads. The only difference is that the panoramic view images run in opposite directions (Fig. 15)[1]. It seems that we only need to build a single model for a scene by a single tour. However, many people have the experiences that we can only recognize a scene well from certain view directions, especially when we go to an environment that we are not so familiar with. We may remember a distinctive landmark (e.g. a T with surrounding buildings) on the way to a destination, but we could be surprised to find that we miss that landmark on the way back after we pass that location.

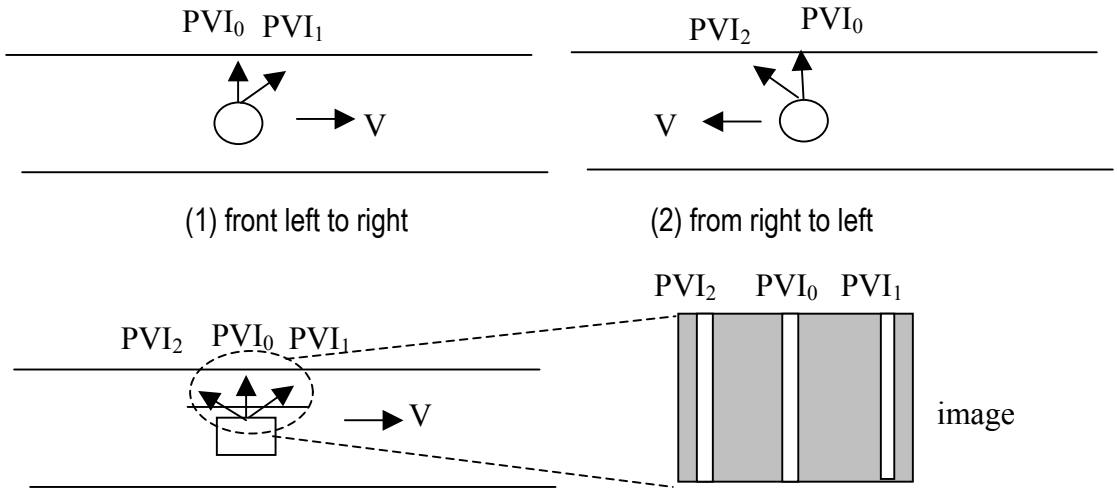


Fig. 15. Viewing directions in generating panoramic view images (PVIs). When an orthogonal viewing direction is used, the panoramic images are the same (PVI_0) for both motion directions. When a forward looking viewing direction is used, the panoramic view images are different (PVI_1 and PVI_2).

The reason is that we almost always need to look forward to know the next landmark in advance. With a forward-looking viewing direction, we may see quite different things from opposite directions. Fig. 16 shows three panoramic view images (PVI_0, PVI_1, PVI_2) generated from the central slit, left slit and right slit of a moving camera as shown Fig. 15(3). It can be seen that in some places that have distinctive depth changes (e.g. in the location marked by a red

[1] We assume that the camera always points to the same road side. For a real application, we can easily using two cameras on a sensor system described in Chapter 5 to build models of both sides of the road scene. The depth offsets in two different lanes of opposite directions can be easily compensated.

rectangle), the appearances of the same scene from the forward-viewing direction (PVI_1) is quite different from those from the backward-viewing direction (PVI_2).

How to solve this problem? A simple solution is to build the landmark model on the orthogonal-looking PVI (Fig. 16(2)), then use temporal context information to predict where is the next landmark before the robot really see it. However, the robot does not make full use of the information in the scene, especially the more distinctive features that vary from different viewing directions. So the natural extension of the panoramic landmark model is to use both the forward-looking PVI and the backward-looking PVI to represent landmarks that will be used in robot localization from two opposite directions.

(1) PVI_2 from the left slit

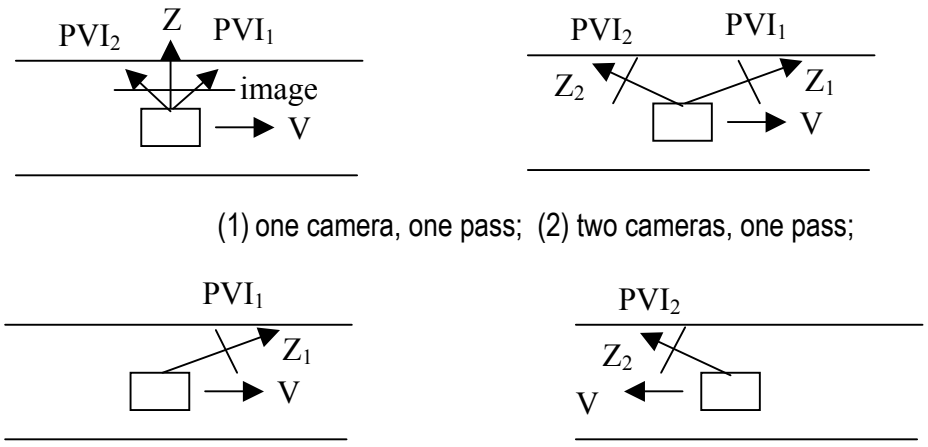
(2) PVI_0 from the central slit

(3) PVI_1 from the right slit

Fig. 16. Panoramic view images (PVIs) from three different viewing directions: backward-looking PVI, orthogonal-looking PVI and forward-looking PVI.

There are several ways to build a bi-PVI representation of a scene. First, we can use a single camera that points to the orthogonal direction of the motion vector (Fig. 17(1)) and generate two PVIs from the left and right slit windows. This only requires on pass to build the scene model; however the viewing direction is limited by the viewing angle of the camera. Second, we can use two cameras pointing to two larger separate directions as shown in Fig. 17(2). In this case only one pass is required to build the scene model, but two cameras are needed. In addition, we need

to perform a image rectification to each frame to generate a rectified image sequence that has a "virtual" optical axis pointing orthogonal to the road side. Finally, we can use one forward-looking camera but run two passes to build the bi-PVI representation of the scene (Fig. 17(3)).



(1) one camera, one pass; (2) two cameras, one pass;

(3) one camera, two passes

Fig. 17. Three ways to build a bi-PVI representation

Hence, the proposed panoramic landmark model is a view-based representation that is built with the road network that the robot will navigate on. The landmarks in this context is the so called "generalized landmarks" that is the salient features (texture, color and depth) in the panoramic view images, and their spatial relations.

2.5. Conclusions and Discussions

Structure from motion is still a hard problem. In order to construct 3D natural scenes from video sequences, we make reasonable constraints on the motion of the camera. However, the motion model is not an ideal one but a practical one that properly describes the motion of an ordinary automobile moving on typical roads. In this manner we generalize the motion of the panoramic view approach and epipolar plane approach from translation (or smooth motion) to a more general and practical outdoor motion: an approximately known translation plus random fluctuations. A systematic approach is proposed to give a full solution from image sequence to large scale and compact 3D model. The proposed two-stage method de-couples fluctuation motion and structure and thus simplifies the structure from motion problem. The panoramic epipolar plane analysis algorithm is more effective than the general ST image analysis methods since only the representative data are processed and the processing for each panoramic epipolar plane can work in parallel. Furthermore, effective methods are proposed and tested for localization of depth/motion boundaries, and interpolation of depth in textureless areas. Image segmentation, feature extraction, and matching are avoided so that the algorithm is fully automatic. It is interesting to note that the same basic algorithms (plus a occlusion and resolution recovery algorithm) can be used to construct 3D model for image-based rendering and synthesized images of arbitrary views can be generated from this model.

While the proposed method is a practical solution for 3D scene modeling, there exist some open problems that need further study. The current algorithm can work well only with dense image sequences with constrained motions; only the modeling of the static scene is studied; errors in the image stabilization stage may propagate to the next stage. More experiments of 3D LAMP construction for high resolution image sequences are needed where methods of depth layering and time re-sampling should be improved. The fusion of depth/motion and spatial structures (textures, edges) also need further study.

Acknowledgments

This work was supported by China Advanced Research Project during 1990-1996, by the China High Technology Program under contract No. 863-306-ZD-10-22 and partially by China National Science Foundation under contact No. 69805003. The first author would like to thank Prof. Edward M. Riseman and Prof. Allen R. Hanson of UMASS at Amherst for their valuable discussions and comments that lead to the final form of this updated English version. Early and short versions of this work appeared at the 1998 IEEE Virtual Reality Annual International Symposium (Zhu et al., 1998a) and the 1999 IEEE Conference on Computer Vision and Pattern Recognition (Zhu et al., 1999b).

Notes

1. This part is discussed (Zhu and Hanson, 2001) under the context of image-based modeling and rendering. For robot navigation, this may not be a necessary step.

2. Camera settings other than this standard stetting are also applicable but an image rectification procedure should be applied first (Zhu, 1997).

3. So in practice, the covariance will be selected adaptively according to the real situation of an ST texture instead of using $\sigma^2 = \frac{m-1}{4}$ directly.

References

Adelson, E. H. and Bergen, J. R. 1985. Spatiotemporal energy model for the perception of motion. *J. Opt. Soc. Am.*, A2: 284-299.

Allmen, M. and Dyer, C. R. 1991. Long range spatiotemporal motion understanding using spatiotemporal flow curves. In *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 303-309.

Baillard, C. and Zisserman, A. 1999. Automatic reconstruction of piecewise planar models from multiple views. In *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 559-565.

Baker, S., Szeliski, R. and Anandan, P. 1998. A layered approach to stereo reconstruction. *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 434-441.

Black, M. J. and Jepson, A. D. 1996. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE Trans Pattern Analysis and Machine Intelligence*, 18(10): 972-986.

Bolles, R. C., Baker, H. H. and Marimont, D. H. 1987. Epipolar-plane image analysis: an approach to determining structure from motion. *Int. J. Computer Vision,* 1(1): 7-55.

Chang, N. L. and Zakhor, A. 1997. View generation for three-dimensional scene from video sequence. *IEEE Trans on Image Processing*, 6(4): 584-598.

Chen, S. E. 1995. QuickTime VR - an image based approach to virtual environment navigation. In *ACM Conf. Proc. SIGGRAPH 95*, pp. 29-38.

Collins, R. 1996. A space-sweep approach to true multi-image matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 358-363.

Collins, R., Jaynes, C., Cheng, Y., Wang, X., Stolle, F., Schultz, H., Hanson, A. and Riseman, E. 1998. The Ascender System: Automated Site Modeling from Multiple Aerial Images," Computer Vision and Image Understanding, 72(2): 143-162.

Coorg, S. and Teller, S. 1998. Automatic extraction of textured vertical facades from pose imagery. *MIT LCS TR-729*.

Dalmia, A. K. and Trivedi, M. 1996. High speed extraction of 3D structure of selectable quality using a translating camera. *Computer Vision and Image Understanding*, 64(1): 97-110.

Debevec, P., Taylor, C. and Malik, J. 1996. Modeling and rendering architecture from photographs: a hybrid geometry- and image- based approach. In *ACM Conf. Proc. SIGGRPAH 96*, pp. 11-20.

Faugeras, O., Robert, L., Laveau, S., Csurka, G., Zeller, C., Gauclin, C. and Zoghlami, I. 1998. 3-D reconstruction of urban scenes from image sequences. *Computer Vision and Image Understanding*, 69(3): 292-309.

Fleet, D. J., Black, M. J., and Jepson, A. D. 1998, Motion feature detection using steerable flow fields. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 274-281.

Freeman, W. T. and Adelson, E. H. 1991. The design and use of steerable filters. *IEEE Trans Pattern Analysis and Machine Intelligence,* 13(9): 891-906.

Hansen, M., Anandan, P., Dana, K., van de Wal, G., and Burt, P., 1994. Real-time scene stabilization and mosaic construction. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 54-62.

Heeger , D. J. 1987. Optical flow from spatio-temporal filters. In *Proc. IEEE Int. Conf. Computer Vision,* pp. 181-190.

Ishiguro, H., Yamamoto, M. and Tsuji S. 1990, Omni-directional stereo for making global map. In *Proc. IEEE Int. Conf. Computer Vision*, pp. 540-547.

Jahne B, Digital Image Processing , Concept, Algorithms and Scientific Applications, Springer-Verlag, 1991

McMillan L. and Bishop, G. 1995. Plenoptic modeling: an image-based rendering system. In *ACM Conf. Proc. SIGGRAPH 95*, pp. 39-46.

Morimoto, C. and Chellappa, R. 1997. Fast 3-D stabilization and mosaic construction. In *IEEE Conf. of Computer Vision and Pattern Recognition*, pp. 660-665.

Murray, D. W. 1995. Recovering range using virtual multicamera stereo, *Computer Vision and Image Understanding.* 61(2): 285-291.

Niyogi, S. A. 1995. Detecting kinetic occlusion. In *Proc. IEEE Int. Conf. Computer Vision*, pp. 1044-1049.

Peleg, S. and Ben-Ezra, M. 1999. Stereo panorama with a single camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 395-401.

Peleg, S. and Herman, J. 1997. Panoramic mosaics by manifold projection. In *IEEE Conference on Computer Vision and Pattern Recognition*: pp. 338-343.

Rademacher, P. and Bishop, G. 1998. Multiple-center-of-projection images. In *Proc. SIGGRAPH'98*, pp. 199-206.

Rousseeuw, P. J. and Leroy, A. M. 1987. *Robust Regression and Outlier Detection*, J. Wiley & Sons, New York.

Sawhney, H. S. and Ayer, S. 1996. Compact representation of videos through dominant and multiple motion estimation. *IEEE Trans Pattern Analysis and Machine Intelligence*, Vol. 18, No. 8, Aug , pp. 814-830.

Sawhney, H. S., Kumar, R., Gendel, G., Bergen, J., Dixon, D. and Paragano, V. 1998. VideoBrush™: Experiences with consumer video mosaicing. In *IEEE Workshop on Application of Computer Vision*, pp. 56-62.

Shade, J., Gortler, S., He. L. and Szeliski, R. 1998. Layered depth image. In *Proc. SIGGRAPH'98,* pp. 231-242.

Shum, H.-Y. and Szeliski, R. 1997. Panoramic Image Mosaics. *Microsoft Research, Technical Report, MSR-TR-97-23.*

Shum, H.-Y., Han, M. and Szeliski, R. 1998. Interactive construction of 3D models from panoramic mosaics. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 427-433.

Shum, H.-Y. and Szeliski, R. 1999. Stereo reconstruction from multiperspective panoramas. In *Proc. IEEE Int. Conf. Computer Vision*, pp. 14-21.

Szeliski, R. 1999. A multi-view approach to motion and stereo. In *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 157-163.

Wang, J. and Adelson, E. H. 1994. Representation moving images with layers. *IEEE Trans. on Image Processing,* 3(5): 625-638.

Xiong, Y. and Turkowski, K. 1997. Creating image-based VR using a self-calibrating fisheye lens. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 237-243.

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

Zheng, J. Y. and Tsuji, S. 1998. Generating Dynamic Projection Images for Scene Representation and Understanding. *Computer Vision and Image Understanding*, 72(3): 237-256.

Zhu, Z. 1997. On environment modeling for visual navigation. *Ph.D. Thesis*, Computer Science Department, Tsinghua University.

Zhu, Z., Xu, G. and Lin, X. 1998a. Constructing 3D natural scene from video sequences with vibrating motions. In *Proc. IEEE Virtual Reality Annual International Symposium (VRAIS-98)*, pp. 105 – 112.

Zhu, Z. Xu, G. Yang, Y. and Jin, J. S. 1998b. Camera stabilization based on 2.5D motion estimation and inertial motion filtering. In *IEEE Intelligent Vehicles Symposium*, Stuttgart, Germany, vol. 2, pp. 329-334.

Zhu, Z., Hanson, A. R., Schultz, H., Stolle F. and Riseman, E. M. 1999a. Stereo Mosaics from a Moving Video Camera for Environmental Monitoring. In *First International Workshop on Digital and Computational Video*, Tampa, Florida, pp. 45-54.

Zhu, Z., Xu, G. and Lin, X., 1999b. Panoramic EPI Generation and Analysis of Video from a Moving Platform with Vibration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 531-537.

Zhu, Z., Xu, G., Riseman, E. M. and Hanson, A. R. 1999c, Fast generation of dynamic and multi-Resolution 360° panorama from video sequences. In *IEEE International Conference on Multimedia Computing and Systems*, Florence, Italy, vol. 1, pp 400-406.

Zhu, A. and Hanson, A. R. 3D LAMP: a New Layered Panoramic Representation, The Eighth IEEE International Conference on Computer Vision, Vancouver, Canada, July 9-12, 2001d.

Zisserman, A., Fitzgibbon, A. and Cross F. 1999. VHS to VRML: 3D graphical models from video sequences. In *IEEE International Conference on Multimedia Computing and Systems*, Florence, Italy, pp. 51-57.