Machine Vision and Applications (2005) 16: 85–95
Digital Object Identifier (DOI) 10.1007/s00138-004-0168-z

Machine Vision and
Applications

Parametric ego-motion estimation for vehicle surround analysis using an omnidirectional camera

Tarak Gandhi, Mohan Trivedi

Computer Vision and Robotics Research Laboratory, University of California at San Diego, La Jolla, CA, USA
(e-mail: {tgandhi,trivedi}@ucsd.edu)

Received: 1 August 2003 / Accepted: 8 July 2004
Published online: 3 February 2005 – © Springer-Verlag 2005

Abstract. Omnidirectional cameras that give a 360° panoramic view of the surroundings have recently been used in many applications such as robotics, navigation, and surveillance. This paper describes the application of parametric ego-motion estimation for vehicle detection to perform surround analysis using an automobile-mounted camera. For this purpose, the parametric planar motion model is integrated with the transformations to compensate distortion in omnidirectional images. The framework is used to detect objects with independent motion or height above the road. Camera calibration as well as the approximate vehicle speed obtained from a CAN bus are integrated with the motion information from spatial and temporal gradients using a Bayesian approach. The approach is tested for various configurations of an automobile-mounted omni camera as well as a rectilinear camera. Successful detection and tracking of moving vehicles and generation of a surround map are demonstrated for application to intelligent driver support.

Keywords: Motion estimation – Panoramic vision – Intelligent vehicles – Driver support systems – Collision avoidance

1 Introduction and motivation

Omnidirectional cameras that give panoramic view of surroundings have become very popular in machine vision. Benosman and Kang [5] give a comprehensive description of panoramic imaging systems and their applications. There is a considerable interest in motion analysis from moving platforms using omni cameras, since panoramic views help in dealing with ambiguities associated with ego-motion of the platforms [15].

In particular, a vehicle surround analysis system that monitors the presence of other vehicles in all directions is important for online as well as offline applications. Online systems are useful for intelligent driver support. On the other hand, offline processing of video sequences is useful for study of behavioral patterns of the driver in order to develop better tools for driver assistance. For such systems, a complete surround analysis system that monitors the lanes and vehicles around the driver is very important. An omni camera mounted on the automobile could provide a complete panoramic view of the surroundings and would be very appropriate for performing such a task. The main contribution of this paper is to perform moving object detection from omni image sequences using a direct parametric motion estimation method and apply it to video sequences obtained from an automobile-mounted camera to detect and track neighboring vehicles.

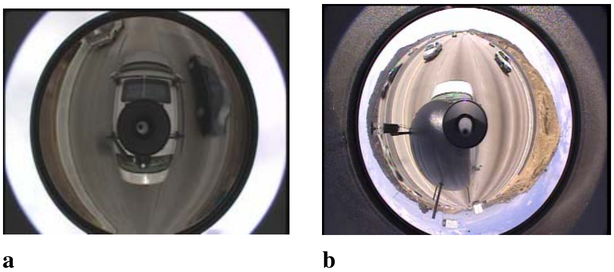

Fig. 1a,b. Images from omni camera mounted on an automobile. **a** This camera has vertical FOV of 5° above the horizon and covers only nearby surroundings but gives larger vehicle images. **b** This camera has a vertical resolution of 15° above the horizon and covers farther surroundings, but with smaller vehicle images

Figure 1 shows the images from omni cameras in different configurations used for this work. It is seen that the camera covers a 360° field of view (FOV) around its center. However, the image it produces is distorted, with straight lines transformed into curves. Directly unwarping the image to perspective image would introduce severe blur in perspective image, causing problems for subsequent steps in motion analysis. Instead, the omni camera transformations are combined with the motion transformations to compensate the ego-motion in the omni domain itself.

1.1 Related work in motion analysis

Motion estimation from moving omni cameras has recently been a topic of great interest. Rectilinear cameras usually have a smaller FOV, due to which the focus of expansion often lies outside the image, causing motion estimation to be sensitive to the camera orientation. Also, the motion field produced by translation in the horizontal direction is similar to that due to rotation about the vertical axis. As noted by Gluckman and Nayar [15], omni cameras avoid both these problems due to their wide FOV. They project the image motion on a spherical surface using Jacobians of transformations to determine ego-motion of a moving platform in terms of translation and rotation of the camera. Vassalo et al. [30] propose a general Jacobian function that can describe a wide variety of omni cameras. Shakernia et al. [26] use the concept of back-projection flow, where the image motion is projected to a virtual curved surface in place of a spherical surface to simplify the Jacobians. Using this concept, they have adapted ego-motion algorithms for rectilinear cameras for use with omni sensors. Svoboda et al. [28] use feature correspondences to estimate the essential matrix between two frames using the 8-point algorithm. They also note that the motion estimation is more stable with omni cameras compared to rectilinear cameras.

Most of these methods first compute motion of image pixels and then use the motion vectors to estimate the motion parameters. However, due to the aperture problem [17], the full motion information is reliable only near cornerlike points. The edge points only have motion information normal to the edge. Direct methods can optimally use the motion information from edges as well as corners to get parameters of motion. Direct methods have often been used with rectilinear cameras for planar motion estimation, obstacle detection, and motion segmentation [7,21,20]. To distinguish objects of interest from extraneous features, the ground is usually approximated by a planar surface whose ego-motion is modeled using a projective transform [25,23] or its linearized version [3]. Using this model, the ego-motion of the ground is compensated in order to separate the objects with independent motion or height.

1.2 Related work on intelligent vehicles

In recent years, considerable research has been conducted on developing intelligent vehicles having driver support systems that enhance safety. Computer vision techniques have been applied to detecting lanes, other vehicles, and pedestrians to warn the driver of dangers such as lane departure and possible collision with other objects.

Stereo cameras are especially useful for detecting obstacles in front that are far from the driver. Bertozzi and Broggi [6] use stereo cameras for lane and obstacle detection. They model the road as a planar surface and use inverse perspective transform to register the road plane between two images. The obstacles above the road would have residual disparity and are easily detected. In the case of curved roads, [24] create a V-disparity image based on clustering similar disparities on each image row. A line or curve in this image corresponds to a straight or curved road, respectively, and the vehicles on the road form other distinctive patterns.

Omni cameras with their panoramic FOV show a great potential in intelligent vehicle applications. In [18], an omni camera mounted inside the car obtained a view of the driver as well as of the surroundings. The driver's pose was estimated using hidden Markov models and was used to generate the driver's view of the surroundings using the same camera. In [2], feature-based methods detecting specific characteristics of vehicles, such as wheels, were used to detect and track vehicles.

Single-camera motion analysis has been used for separating ego-motion of the background to detect vehicles and other obstacles on the road. Robust real-time motion compensation for the road plane for this purpose is described in [23]. In [10], a system for video-based driver assistance involving lane and obstacle detection using a rectilinear camera is described. Direct parametric motion estimation discussed in the previous section is especially useful for vehicle applications, since most of the features on the road are line-based and very few corner features are available. The direct estimation approach was generalized for motion compensation using omni cameras in [13,18], where parameters of planar homography were estimated. A modification of that approach is used here as in [14] to estimate the vehicle ego-motion in terms of linear and angular velocities. These are used to compensate the ego-motion for the road plane and detect vehicles having residual motion to generate a complete surround view showing the position and tracks of the vehicles.

2 Ego-motion estimation and compensation system

The system block diagram is shown in Fig. 2. The inputs to the system are a sequence of images from an omni camera mounted on automobile, the vehicle speed from the CAN bus which gives information about the vehicle state, and the nominal calibration of the camera with respect to the road plane. The state of the vehicle containing vehicle velocity and calibration are used to compute the warping parameters to compensate the image motion between two frames for points on the road plane. The warping transform is a composition of the omni camera transform and the planar motion model. It transforms the omni image coordinates to perspective coordinates, applies the planar motion parameters to compensate the road motion, and converts them back to the omni view. Two consecutive frames from the image sequence are taken, and the warping parameters are used to transform one image to another, to compensate the motion of the road as much as possible. The objects with independent motion and height would have large residual motion, making it possible to separate them from road features.

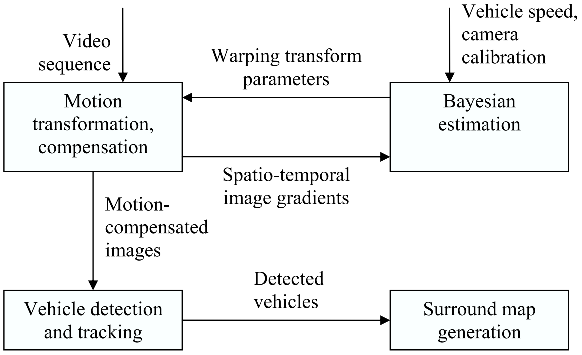
Video sequence
Warping transform parameters
Vehicle speed, camera calibration
Motion transformation, compensation
Bayesian estimation
Spatio-temporal image gradients
Motion-compensated images
Detected vehicles
Vehicle detection and tracking
Surround map generation

Fig. 2. System for ego-motion compensation from a moving platform. The inputs to the system are the video sequence from an omni camera and vehicle speed information extracted from the CAN bus of the car that provides a number of variables of the car's dynamics. The output is a surround map with detected vehicles and their tracks

However, the features on the road may also have some residual motion due to errors in the vehicle speed and calibration parameters. To correct for these errors, spatial and temporal gradients of the motion-compensated images are obtained. Bayesian estimation similar to [23] is applied with gradients as observations to update the prior knowledge of the state of the vehicle using Kalman filter measurement update equations. To minimize the effect of outliers, only the gradients satisfying a constraint on the residual are used in the estimation process. The updated vehicle state is used to recompute the warping parameters, and the residual gradients are recomputed. The process is repeated in a coarse-to-fine iterative manner. The gradients computed using the finally updated state of the vehicle are used to separate the vehicle features from the road features. The vehicle features are combined using constraints on vehicle length and separation to obtain blobs corresponding to vehicles that are tracked over the number of frames. The surround map is generated by unwarping the omni image to give a plan view and superimposing the vehicle blobs and tracks over the resulting image. The following sections describe the processing steps in detail.

3 Motion transformations for omni camera

Let c denote a nominal camera coordinate system, based on the known camera calibration, with the Z-axis along the camera axis and the $X - Y$ plane being the imaging plane. Due to camera vibrations and drift, the actual camera system at any given time is assumed to have small rotation with respect to this system due to vibrations and drift. Use of the nominal system allows us to treat small rotations as angular displacement vectors. The ego-motion of the camera is then described using state vector $\mathbf{x}$ containing the camera linear velocity V, angular velocity W, and angular displacement A between nominal camera system c and actual system a, all expressed in a nominal camera system c.

3.1 Planar motion model

To detect obstacles in the path of a moving camera, the road is modeled as a planar surface. Let P_a and P_b denote the perspective projections of a point on the road plane in coordinate systems corresponding to two positions a and b of the moving camera. These are related by:

$$\lambda_b P_b = \lambda_a R P_a + D_b^a = \lambda_a \left[R P_a + D / \lambda_a \right] , \qquad (1)$$

where R and D denote the rotation and translation between the camera positions, and λ_a, λ_b depend on the distance of the actual 3D point. Let the equation of the road plane at the camera position a be:

$$K^T (\lambda_a P_a) = 1 \, , \qquad (2)$$

where K is vector normal to the road plane in the coordinate system of camera position a. Substituting the value of λ_a from Eq. 2 in Eq. (1), it is seen that P_a and P_b are related by a projective transform [11]:

$$\lambda_b P_b = \lambda_a \left[R + D K^T \right] P_a = \lambda_a H P_a \, , \qquad (3)$$

where $H = R + DK^T$ is known as the projective transform or homography. This relation has been widely used to estimate planar motion for rectilinear cameras.

If the angular displacements with respect to the nominal camera calibration are small, the matrices can be expressed as:

$$\begin{aligned} R &\simeq I - W_\times \Delta t \, , \\ D &\simeq - \left[I - W_\times \Delta t - A_\times \right] V \Delta t \, , \\ K &\simeq \left[I - A_\times \right] K_0 \, , \end{aligned} \qquad (4)$$

where $W_\times$ and $A_\times$ represent the skew symmetric matrices constructed from vectors W and A, and K_0 represents the plane normal in the nominal camera coordinate system.

3.2 Omni camera transform

To apply the ego-motion estimation method to omni cameras, one needs the mapping from the camera coordinate system to the pixel domain and vice versa. Given this transformation and the planar motion model, one can generate a transformation that compensates the motion of the planar surface in the omni pixel domain.

In particular, the omni camera used in this work consists of a hyperbolic mirror and a camera placed on its axis, with the center of projection of the camera on one of the focal points of the hyperbola. It belongs to a class of cameras known as central panoramic catadioptric cameras [5]. These cameras have a single viewpoint that permits the image to be suitably transformed to obtain perspective views.

The geometry of a hyperbolic omni camera is shown in Fig. 3a. According to the mirror geometry, a light ray from the object toward the viewpoint at the first focus O is reflected so that it passes through the second focus, where a conventional rectilinear camera is placed. The equation of the hyperboloid is given by:

$$\frac{(Z-c)^2}{a^2} - \frac{X^2 + Y^2}{b^2} = 1 \, , \qquad (5)$$

where $c = \sqrt{a^2 + b^2}$.

Let $P = (X, Y, Z)^T$ denote the homogenous coordinates of the perspective transform of any 3D point λP on ray OP, where λ is the scale factor depending on the distance of the 3D point from the origin. It can be shown [1,19,26] that the reflection in the mirror gives the point $-p = (-x, -y)^T$ on the image plane of the camera using:

$$p = \begin{pmatrix} x \\ y \end{pmatrix} = \frac{q_1}{q_2 Z + q_3 \|P\|} \begin{pmatrix} X \\ Y \end{pmatrix} \, , \qquad (6)$$

where

$$q_1 = c^2 - a^2 , \; q_2 = c^2 + a^2 , \; q_3 = 2ac , \qquad (7)$$

and $\|P\| = \sqrt{X^2 + Y^2 + Z^2}$.

Note that the expression for image coordinates p is independent of the scale factor λ. The pixel coordinates $w = (u, v)^T$ are then obtained by using the calibration matrix K of the conventional camera composed of the focal lengths f_u, f_v, optical center coordinates $(u_0, v_0)^T$, and camera skew s, or:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} . \qquad (8)$$

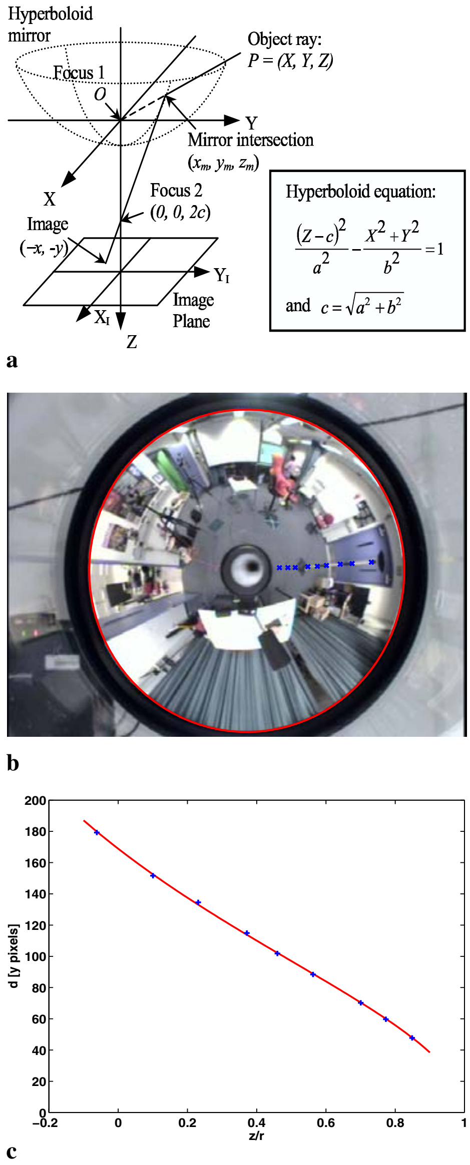

Fig. 3. **a** Geometry of a hyperbolic omni camera. The rays toward the first focus of the mirror are reflected toward the second focus and imaged by a normal camera. **b** FOV of omni camera with number of points with known coordinates. **c** Curve fitting for internal parameter estimation

This transform can be used to warp an omni image to a plan perspective view. To convert perspective view back to omni view within a scale factor, the inverse transformation can be used:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = K^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} , \tag{9}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \equiv \begin{pmatrix} q_1 x \\ q_1 y \\ q_2 - q_3 \sqrt{x^2 + y^2 + 1} \end{pmatrix} . \tag{10}$$

It should be noted that the transformation of omni to perspective view involves very different magnifications in different parts of the image. For this reason, the quality of the image deteriorates if the entire image is transformed at one time. Hence, as noted by Daniilidis [8], it is desirable to perform motion estimation directly in the omni domain, but use the above transformations to map the locations to the perspective domain as required.

Since the internal parameters of the omni camera are to be measured only once, a specialized setup was used to obtain the calibration. The omni camera was set on a tripod and leveled to have vertical camera axis. A number of features with known coordinates were taken on the ground and a vertical pole to cover the FOV of the omni camera. The FOV covered by the omni camera maps into the ellipse as seen in Fig. 3a. The camera center and aspect ratio were computed from the ellipse parameters. Using these parameters, the image coordinates (u, v) can be normalized to give (u', v') corresponding to origin as center and unit aspect ratio. Assuming radial symmetry around the image center, we have:

$$d = \sqrt{u'^2 + v'^2} = \frac{\sqrt{X^2 + Y^2}}{c_1 Z + c_2 \|P\|} , \tag{11}$$

where $c_1 = q_2/(q_1 f_v)$ and $c_2 = q_3/(q_1 f_v)$. Using the known world and image coordinates of these points, the linear equations in c_1 and c_2 are formed and solved using least squares:

$$dZc_1 + d\|P\|c_2 = \sqrt{X^2 + Y^2} . \tag{12}$$

Figure 3b shows the plot of d against $Z/\|P\|$ of the sample points and the curve fitted using estimated parameters. It is seen that the curve models the omni mapping quite faithfully. Nonlinear least squares can then be used for improving the accuracy.

Though the method is designed for central panoramic cameras, if the distance to the observed scene is large compared to the mirror size, the method can also be applied to noncentral panoramic cameras provided the mapping from object ray directions to pixel coordinates is known. In fact, it was observed that for a hyperbolic mirror, the FOV is concentrated within a close distance around the camera, which made it somewhat difficult to detect objects farther from the camera where resolution was poor. Noncentral cameras may be particularly useful since they give more flexibility in adjusting the camera resolution in different parts of the image, as described in [16].

4 Ego-motion estimation

To estimate the ego-motion parameters, the parametric image motion is substituted into the optical flow constraint [17]:

$$g_u \Delta u + g_v \Delta v + g_t = 0 , \tag{13}$$

where g_u, g_v are spatial gradients and g_t is the temporal gradient. Since the image motion $(\Delta u, \Delta v)$ at each point i can be represented as a function of the incremental state vector $\Delta\mathbf{x}$, the optical flow constraint Eq. 13 for image points $1 \ \ldots \ N$ can be expressed as:

$$\Delta\mathbf{z} = \mathbf{c}(\Delta\mathbf{x}) + \mathbf{v} \simeq \mathbf{C}\Delta\mathbf{x} + \mathbf{v} , \tag{14}$$

Table 1. Chain of functions and Jacobians leading from state vector $\mathbf{x}$ to optical flow constraint $\mathbf{c}$. Rows 4 and 5 correspond to the omni camera transform that converts the camera coordinates to pixel coordinates

$\mathbf{x} = \begin{pmatrix} V \\ W \\ A \end{pmatrix}$	$H = R + DK^T$	$\partial H = \partial R + \partial D.K^T + D(\partial K)^T$
$H = \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix}$	$R \simeq I - W_\times \Delta t$ $D \simeq [I - A_\times] V \Delta t$ $K \simeq [I - A_\times] K_0$	$\partial R = \partial W_\times \Delta t$ $\partial D = (I - W_\times \Delta t - A_\times)\Delta t \partial V - (\partial W_\times \Delta t + \partial A_\times) V \Delta t$ $\partial K = -A_\times K_0$ $\partial V / \partial V_i = e_i,\ \partial W_\times / \partial W_i = \partial A_\times / \partial A_i = (e_i)_\times$
$h = \begin{pmatrix} h_1 \dots h_9 \end{pmatrix}^T$	$\begin{pmatrix} X_b \\ Y_b \\ Z_b \end{pmatrix} \equiv \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix} \begin{pmatrix} X_a \\ Y_a \\ Z_a \end{pmatrix}$	$\frac{\partial P_b}{\partial h} = \begin{pmatrix} X_a & Y_a & Z_a & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X_a & Y_a & Z_a & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X_a & Y_a & Z_a \end{pmatrix}$
$P = \begin{pmatrix} X & Y & Z \end{pmatrix}^T$	$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} = \frac{q_1}{q_2 Z + q_3 \lVert P \rVert} \begin{pmatrix} X \\ Y \end{pmatrix}$	$\frac{\partial p}{\partial P} = \frac{1}{(q_2 Z + q_3 \lVert P \rVert) \lVert P \rVert} \cdot \begin{pmatrix} q_3 x X - q_1 \lVert P \rVert & q_3 x Y & q_3 x Z \\ q_3 y X & q_3 y Y - q_1 \lVert P \rVert & q_3 y Z \end{pmatrix}$
$p = \begin{pmatrix} x & y \end{pmatrix}^T$	$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$	$\frac{\partial w}{\partial p} = \begin{pmatrix} f_u & s \\ 0 & f_v \end{pmatrix}$
$w = \begin{pmatrix} u & v \end{pmatrix}^T$	$\mathbf{c} = \begin{pmatrix} g_u & g_v \end{pmatrix} \begin{pmatrix} u_b - u_a \\ v_b - v_a \end{pmatrix} = -g_t + \eta$	$\frac{\partial \mathbf{c}}{\partial w_b} = \begin{pmatrix} g_u & g_v \end{pmatrix}$

where

$$\mathbf{c}(\Delta \mathbf{x}) = \begin{pmatrix} (g_u \Delta u + g_v \Delta v)_1 \\ \vdots \\ (g_u \Delta u + g_v \Delta v)_N \end{pmatrix}, \tag{15}$$

$$\Delta \mathbf{z} = -\begin{pmatrix} (g_t)_1 \\ \vdots \\ (g_t)_N \end{pmatrix},$$

$\mathbf{v}$ is the vector of measurement noise in the time gradients, and $\mathbf{C} = \partial \mathbf{c} / \partial \mathbf{x}$ is the Jacobian matrix computed using the chain rule as in [13]. The function $\mathbf{c}(\mathbf{x})$ is nonlinear. Row i of the $N \times 9$ Jacobian matrix is given by the chain rule:

$$\mathbf{C}_i = \left(\frac{\partial \mathbf{c}_i}{\partial \mathbf{x}} \right) = \left(\frac{\partial \mathbf{c}}{\partial w_b} \frac{\partial w_b}{\partial p_b} \frac{\partial p_b}{\partial P_b} \frac{\partial P_b}{\partial h} \frac{\partial h}{\partial \mathbf{x}} \right)_i, \tag{16}$$

where $P_b = (X_b, Y_b, Z_b)^T$, $p_b = (x_b, y_b)^T$ and $w_b = (u_b, v_b)^T$ are the coordinates of the point in the camera, image, and pixel coordinate systems for camera position b, and h is the vector of elements of H. The individual Jacobians are computed similarly to [13]. The relationship between these variables, and their Jacobians, are shown in Table 1.

Since the points having very low texture do not contribute much to the estimation of motion parameters, only those image points having gradient magnitude above a threshold value are selected for performing estimation. Alternatively, a nonmaximal suppression is performed on the image gradients and the image points with local maxima are used. This way, instead of computing Jacobians using multiple image transforms over the entire image, the Jacobians are computed only at the selected points that have significant information for estimating parameters.

The estimates of the state $\mathbf{x}$ and its covariance $\mathbf{P}$ are iteratively updated using the measurement update equations of the iterated extended Kalman filter [4]:

$$\mathbf{P} \leftarrow \left[\mathbf{C}^T \mathbf{R}^{-1} \mathbf{C} + \mathbf{P}_{-}^{-1} \right]^{-1} \tag{17}$$

$$\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + \Delta \hat{\mathbf{x}} = \hat{\mathbf{x}} + \mathbf{P} \left[\mathbf{C}^T \mathbf{R}^{-1} \Delta \mathbf{z} - \mathbf{P}_{-}^{-1} (\hat{\mathbf{x}} - \mathbf{x}_{-}) \right]. \tag{18}$$

However, the optical flow constraint equation is satisfied only for small image displacements up to 1 or 2 pixels. To estimate larger motions, a coarse-to-fine pyramidal framework [22,27] is used. In this framework, a multiresolution Gaussian pyramid is constructed for adjacent images in the sequence. The motion parameters are first computed at the coarsest level, and the image points at the next finer level are warped using the computed motion parameters. The residual motion is computed at the finer level, and the process is repeated until the finest level.

Note that since the resolution of the mirror is not constant, formation of Gaussian pyramid could have errors in the neighborhood. However, since the pyramid is used iteratively in coarse-to-fine manner, the errors at lower resolution are expected to be corrected at higher resolution.

The parameters can also be updated from frame to frame using time update equations of the Kalman filter:

$$\hat{\mathbf{x}} \leftarrow \mathbf{B}\hat{\mathbf{x}},\ \mathbf{P} \leftarrow \mathbf{B}\mathbf{P}\mathbf{B}^{\mathbf{T}} + \mathbf{Q}, \tag{19}$$

where $\mathbf{B}$ and $\mathbf{Q}$ are determined from system dynamics.

4.1 Outlier removal

The above estimate is optimal only when all points really belong to the planar surface and when the underlying noise distributions are Gaussian. However, the estimation is highly sensitive to the presence of outliers, i.e., points not satisfying the road motion model. These features should be separated using a robust method. For this purpose, first the region of interest of road is determined using calibration information, and the processing is done only in that region to avoid extraneous features. To detect outliers, an approach similar to the data snooping approach discussed in [9] has been adapted for Bayesian estimation. In this approach, the error residual of each feature is compared with the expected residual covariance at every iteration, and the features are reclassified as inliers or outliers.

If a point $\mathbf{z}_i$ is not included in the estimation of $\hat{\mathbf{x}}$ – i.e., is currently classified as an outlier – then the covariance of its residual is:

$$\begin{aligned} V\left[\boldsymbol{\Delta}\mathbf{z}_i - \mathbf{C}_\mathbf{i}\boldsymbol{\Delta}\hat{\mathbf{x}}\right] &= V\left[\boldsymbol{\Delta}\mathbf{z}_i\right] + \mathbf{C}\,V\left[\hat{\mathbf{x}}\right]\,\mathbf{C}^T \\ &= \mathbf{R} + \mathbf{C}_i\mathbf{P}\mathbf{C}_i^T\,. \end{aligned} \tag{20}$$

However, if $\mathbf{z}_i$ is included in the estimation of $\hat{\mathbf{x}}$ – i.e., is currently classified as an inlier – then it can be shown that the covariance of its residual is given by:

$$V\left[\boldsymbol{\Delta}\mathbf{z}_i - \mathbf{C}_\mathbf{i}\boldsymbol{\Delta}\hat{\mathbf{x}}\right] = \mathbf{R} - \mathbf{C}_i\mathbf{P}\mathbf{C}_i^T < \mathbf{R}\,. \tag{21}$$

Hence, to classify in the next iteration, the residual is compared with its covariance according to whether it is currently an outlier or an inlier. If the Mahalanobis norm is greater than a given threshold, the point is classified as outlier, otherwise as an inlier.

Alternatively, robust-M estimation [12] could be used to reduce the effect of outliers by iteratively reweighting the contribution of samples according to their error residuals.

4.2 Algorithm for motion parameter estimation

The algorithm for iterative estimation of motion parameters is described below:

- Form a Gaussian pyramid from images A and B from consecutive frames.
- Set the initial parameters and the covariance matrix to their priors as: $\hat{\mathbf{x}} = \mathbf{x}_-$ and $\mathbf{P} = \mathbf{P}_-$.
- Proceeding from the coarsest to the finest level, perform multiple iterations of the following steps:
  1. Warp image B using current estimate $\hat{\mathbf{x}}$ of motion parameters to form image $W(B;\hat{\mathbf{x}})$.
  2. Obtain spatial and temporal gradients between image A and the warped image $W(B;\hat{\mathbf{x}})$.
  3. Use optical flow constraint with parametric motion model on inlier points to apply incremental correction in motion parameters and their covariances according to Eqs. 17 and 18.
  4. Compare the residuals of all points with their expected covariances in Eqs. 20 and 21 to reclassify them as inliers and outliers.

5 Vehicle detection and tracking

After motion compensation, the features on the road plane would be aligned between the two frames, whereas those due to obstacles would be misaligned. Image difference between the frames would therefore enhance the obstacles and suppress the road features. To reduce the dependence on local texture, the normalized frame difference [29] is used. This is given at each pixel by:

$$\frac{\langle g_t\sqrt{g_u^2 + g_v^2}\rangle}{k + \langle g_u^2 + g_v^2\rangle}\,, \tag{22}$$

where g_u, g_v are spatial gradients, g_t is the temporal gradient after motion compensation, and $\langle\cdot\rangle$ denotes a Gaussian weighted averaging performed over a $K \times K$ neighborhood of each pixel. In fact, the normalized difference is a smoothed version of the normal optical flow and hence depends on the amount of motion near the point.

Due to the untextured interior of a vehicle, blobs are usually detected at the sides of the vehicle. To get the full vehicle, it is assumed that if two blobs are within a threshold distance (5.0 m) in the direction of the car's motion, they constitute a vehicle. To detect this situation, the original image is unwarped using the flat plane transform, and a morphological closing is performed on the transformed image using a $1 \times N$ vertical mask.

After the blobs corresponding to moving objects are identified, nearby blobs are clustered and tracked over frames using Kalman filtering [4]. The points on the blob that are nearest to the camera center usually correspond to the road plane and are marked as an obstacle map. The vehicle position on the road is computed by projecting the track location on the obstacle map. Since the obstacle map is assumed to be on the road plane, the location of the vehicle can be obtained by inverse perspective transform.

6 Experimental studies

The ego-motion compensation approach was applied for detecting vehicles from an omni camera mounted on an automobile testbed used for intelligent vehicle research. The testbed is instrumented with a number of cameras and computers to capture synchronized video of the surroundings. In addition, the CAN bus of the vehicle gives information on vehicle speed, pedal and brake positions, radar, etc. The vehicle was driven on a freeway as well as on city roads. The maximum vehicle speed for the test was 65 miles per hour (29 m/s). The actual vehicle speed, obtained from the CAN bus, was used for initial motion estimate.

The first test run was conducted with an omni camera having a vertical FOV of only 5° above the horizon. For this reason, only the vehicles near the car were observed, but the resolution was as large as possible. To get as little of the car as possible, the camera was raised 18 in. (45 cm) above the car using a specially designed fixture. Figure 4a shows an image from the omni camera on the car being driven on the freeway. The estimated parametric motion is shown using red arrows. Note that the motion is estimated only in the designated region of interest, which excludes the car body. Figure 4b shows the classification of points into inliers (gray), outliers (white), and unused (black) points. The estimation is done using only the inlier points. An image with the normalized frame difference between the motion-compensated frames is shown in Fig. 4c, which enhances the regions corresponding to independently moving vehicles. Figure 4d shows the detection and tracking of vehicles marked with track ID and the coordinates in the road plane. The omni image was transformed to obtain

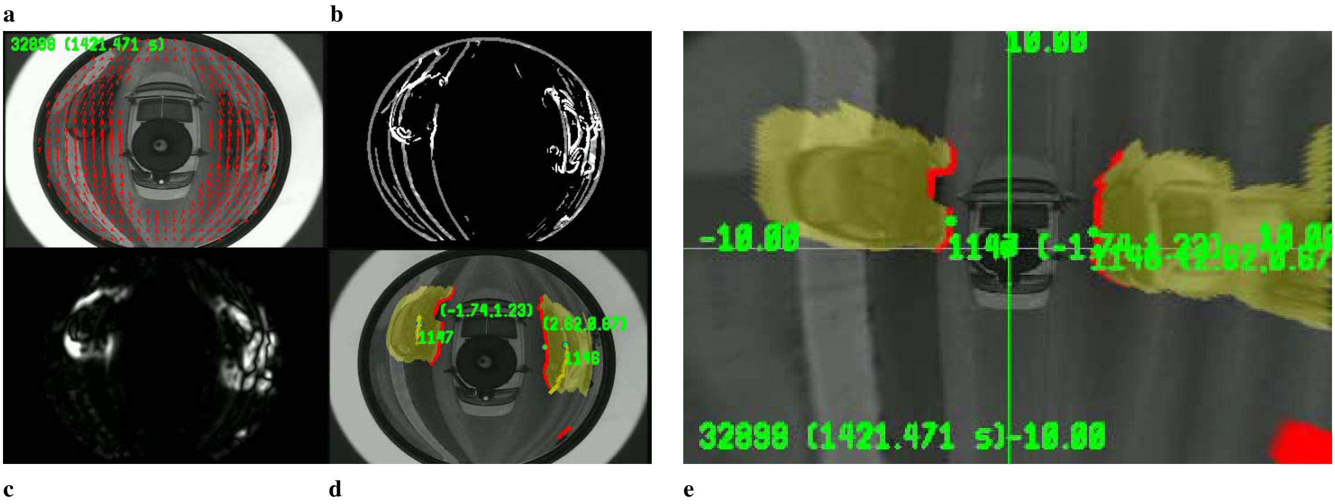

Fig. 4. **a** Image from a sequence using an omni camera mounted on a moving car with estimated parametric motion of road plane. **b** Classification of points into inliers (*gray*), outliers (*white*), and unused (*black*). **c** Normalized difference between motion-compensated images. **d** Detection and tracking of moving vehicles marked with track ID and the coordinates in road plane. **e** Surround view generated by transforming the omni image

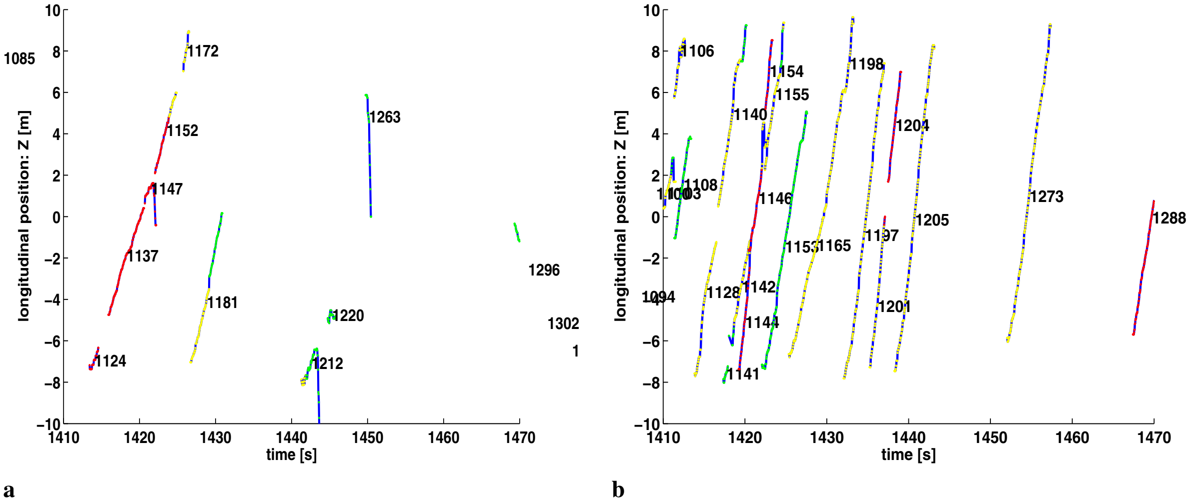

Fig. 5. Plot of the longitudinal position of vehicle tracks on two sides of the car against time. The tracks are color coded as *red, yellow*, and *green* in order of increasing lateral distance from the camera

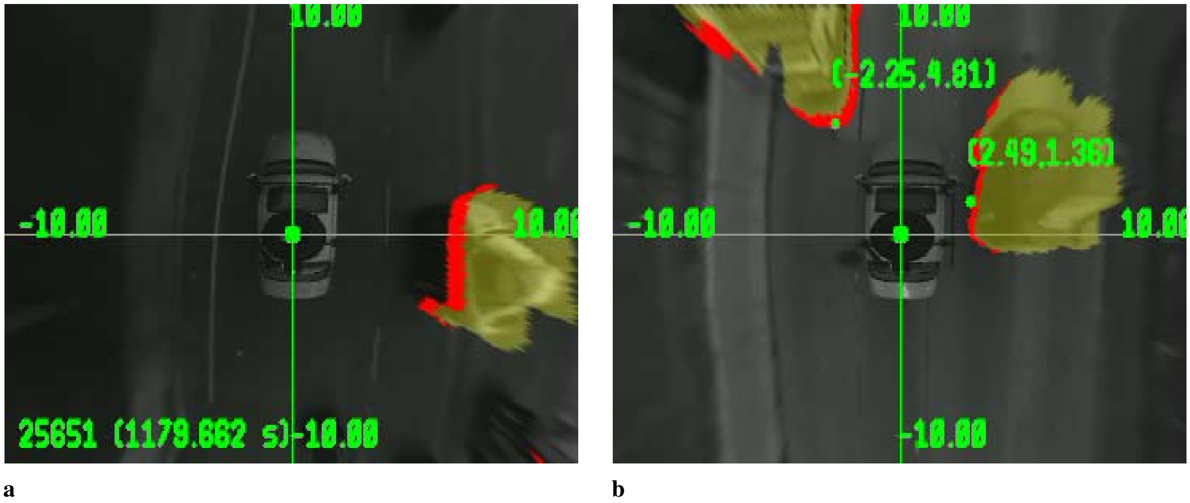

Fig. 6. Surround analysis in different situations with top-mounted camera. **a** City road. **b** Freeway

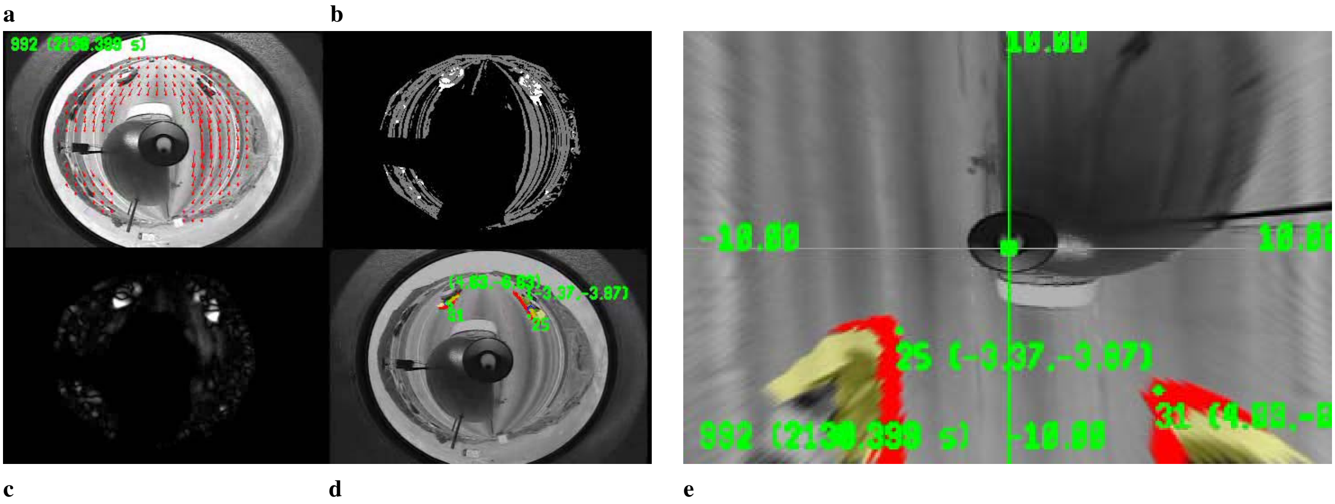

Fig. 7. **a** Image from a sequence using an omni camera with wider FOV mounted on a moving car. The range of the camera is increased but the resolution is decreased. **b** Classification of points into inliers (*gray*), outliers (*white*), and unused (*black*). **c** Normalized difference between motion-compensated images. **d** Detection and tracking of moving vehicles marked with track ID and the coordinates in road plane. **e** Surround view generated by dewarping omni image

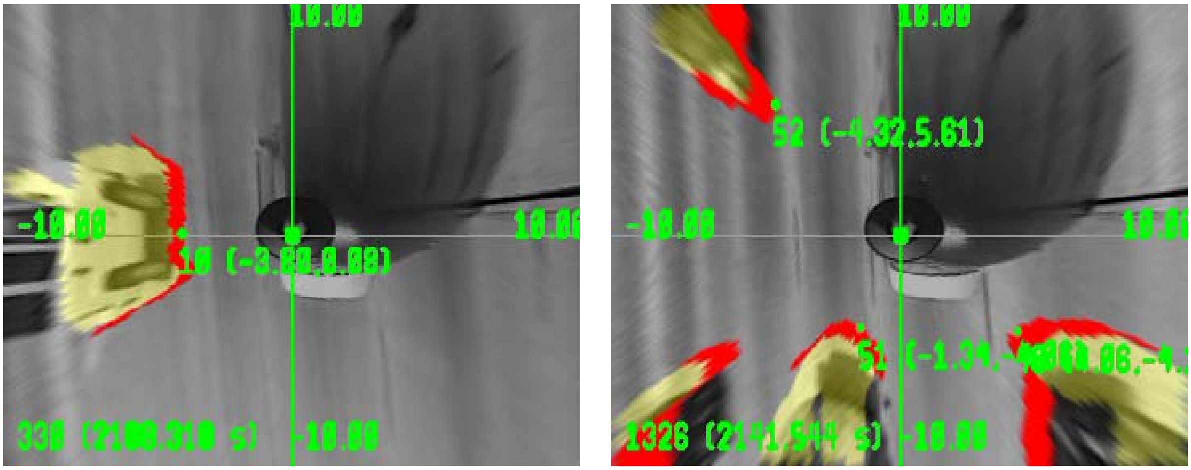

Fig. 8. Samples showing surround vehicle detection with wider FOV omni camera

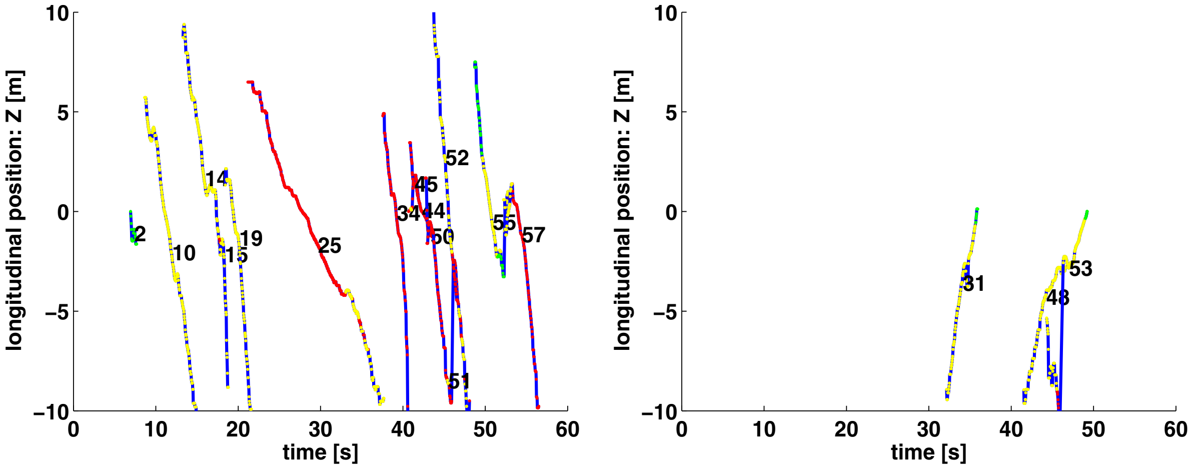

Fig. 9. Plot of longitudinal position of vehicle tracks against time on two sides of a car against time. The tracks are color coded as *red, yellow*, and *green* in order of increasing lateral distance from the camera

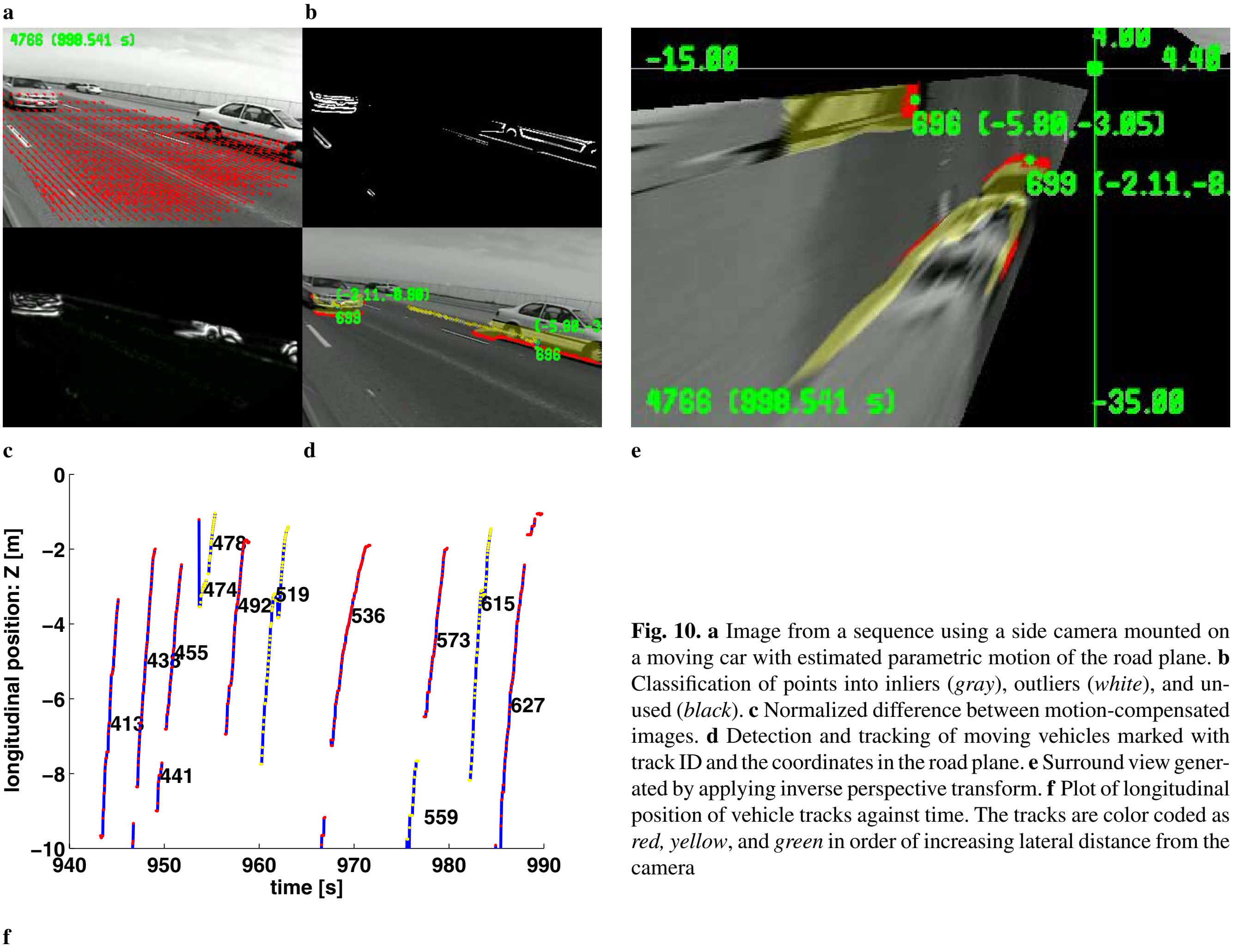

Fig. 10. **a** Image from a sequence using a side camera mounted on a moving car with estimated parametric motion of the road plane. **b** Classification of points into inliers (*gray*), outliers (*white*), and unused (*black*). **c** Normalized difference between motion-compensated images. **d** Detection and tracking of moving vehicles marked with track ID and the coordinates in the road plane. **e** Surround view generated by applying inverse perspective transform. **f** Plot of longitudinal position of vehicle tracks against time. The tracks are color coded as *red, yellow*, and *green* in order of increasing lateral distance from the camera

the plan view of the car surround, as shown in Fig. 4e. The longitudinal position of the car with reference to camera was recorded for each track. Figure 5 shows the plots of track positions against time separately for vehicles on two sides of the camera. The test run also contained sections driven on city roads that had lane marks and other features that were more prominent compared to the freeway. Figure 6 shows examples of moving vehicle detection on city roads as well as in freeway conditions.

The second test run was conducted using an omni camera with FOV 15° above the horizon. It was noted that the camera can see vehicles at a greater distance than the previous camera. The tradeoff was a lower resolution due to which the vehicles had a smaller image size, making them slightly more difficult to detect. Figure 7 shows the result of surround vehicle detection at a larger longitudinal distance from the camera. Figure 8 shows more examples of vehicle detection. Figure 9 shows the plots of track positions against time separately for vehicles on two sides of the camera.

It should be noted that the simplified version of the surround analysis algorithm developed in this paper can also be used with commonly available rectilinear cameras. We conducted several experiments where video streams were acquired using a rectilinear camera mounted on a car window to get a rear side view on the driver's side. Figure 10 shows the results of the detection algorithm. Figure 10e shows the top view generated by applying the inverse perspective transformation using the known calibration. Instead of the full surround view, which can be acquired using an omni camera, only a partial view on one side of the vehicle was obtained.

7 Summary

This paper described an approach to object detection using ego-motion compensation from automobile-mounted omni cameras using direct parametric motion estimation. The road was modeled as a planar surface, and the equations for planar motion transform were combined with the omni camera transform. An optical flow constraint was used to optimally combine the prior knowledge of ego-motion parameters with the information in the image gradients. Coarse-to-fine motion estimation was used, and the motion between the frames was compensated at each iteration. Experimental results demonstrated vehicle detection in two different configurations of omni cameras that obtain near and far views of the surround.

Acknowledgements. This research was supported by a UC Discovery Program digital media grant in collaboration with the Nissan Research Center. We also thank our colleagues from the CVRR Laboratory for their contributions and support. We also thank Dr. Erwin Boer for his suggestions on visualizing the results. Finally we thank the reviewers for their insightful comments which helped us to improve the quality of the paper.

Author, please note. The list of References in your data is not exactly the same as that in your printout. Please check carefully. Thank you.

References

1. Achler O, Trivedi MM (2002) Real-time traffic flow analysis using omnidirectional video network and flatplane transformation. In: Workshop on intelligent transportation systems. Chicago
2. Achler O, Trivedi MM (2004) Vehicle wheel detector using 2d filter banks. In: Proc. IEEE symposium on intelligent vehicles, pp 25–30
3. Adiv G (1985) Determining three-dimensional motion and structure from optical flow generated by several moving objects. IEEE Trans Pattern Anal Mach Intell 7(4):384–401
4. Bar-Shalom Y, Li XR, Kirubarajan T (2001) Estimation with applications to tracking and navigation. Wiley, New York
5. Benosman R, Kang SB (2001) Panoramic vision: sensors, theory, and applications. Springer, Berlin Heidelberg New York
6. Bertozzi M, Broggi A (1998) Gold: a parallel real-time stereo vision system for generic obstacle and lane detection. IEEE Trans Image Proc 7(1):62–81
7. Black MJ, Anandan P (1996) The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. Comput Vis Image Understand 63(1):75–104
8. Daniilidis K, Makadia A, Bulow T (2002) Image processing in catadioptric planes: spatiotemporal derivatives and optical flow computation. In: IEEE workshop on omnidirectional vision, pp 3–12
9. Danuser G, Stricker M (1998) Parametric model fitting: from inlier characterization to outlier detection. IEEE Trans Pattern Anal Mach Intell 20(2):263–280
10. Enkelmann W (2001) Video-based driver assistance: from basic functions to applications. Int J Comput Vis 45(3):201–221
11. Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT Press, Cambridge, MA
12. Forsyth D, Ponce J (2003) Computer vision: a modern approach. Prentice-Hall, Upper Saddle River, NJ
13. Gandhi T, Trivedi MM (2003) Motion analysis of omnidirectional video streams for a mobile sentry. In: 1st ACM international workshop on video surveillance, pp 49–58, Berkeley, CA
14. Gandhi T, Trivedi MM (2004) Motion based vehicle surround analysis using omni-directional camera. In: Proc. IEEE symposium intelligent vehicles, pp 560–565
15. Gluckman J, Nayar S (1998) Ego-motion and omnidirectional cameras. In: Proc. international conference on computer vision, pp 999–1005
16. Hicks RA, Bajcsy R (1999) Reflective surfaces as computational sensors. In: Proc. 2nd workshop on perception for mobile agents, pp 82–86
17. Horn B, Schunck B (1981) Determining optical flow. In: DARPA81, pp 144–156
18. Huang, K, Trivedi MM, Gandhi T (2003) Driver's view and vehicle surround estimation using omnidirectional video stream. In: IEEE symposium intelligent vehicles, Columbus, OH, pp 444–449
19. Huang KC, Trivedi MM (2003) Video arrays for real-time tracking of persons, head and face in an intelligent room. Mach Vis Appl 14(2):103–111
20. Irani M, Anandan P (1998) A unified approach to moving object detection in 2D and 3D scenes. IEEE Trans Pattern Anal Mach Intell 20(6):577–589
21. Irani M, Rousso B, Peleg S (1994) Computing occluding and transparent motions. Int J Comput Vis 12:5–16
22. Jähne B, Haußecker H, Geißler P (1999) Handbook of Computer Vision and Applications, vol 2, chap 14, pp 397–422. Academic Press, San Diego, CA
23. Kruger W (1999) Robust real time ground plane motion compensation from a moving vehicle. Mach Vis Appl 11:203–212
24. Labayrade R, Aubert D, Tarel J-P (2002) Real time obstacle detection in stereovision on non flat road geometry through v-disparity representation. In: IEEE symposium intelligent vehicles, 2:646–651
25. Lourakis MIA, Orphanoudakis SC (1998) Visual detection of obstacles assuming a locally planar ground. In: Asian conference on computer vision, 2:527–534
26. Shakernia O, Vidal R, Sastry S (2003) Omnidirectional egomotion estimation from back-projection flow. In: IEEE workshop on omnidirectional vision
27. Simoncelli EP (1993) Coarse-to-fine estimation of visual motion. In: Proc. 8th workshop on image and multidimensional signal processing, Cannes, France, pp 128–129
28. Svoboda T, Pajdla T, Hlaváč V (1998) Motion estimation using central panoramic cameras. In: IEEE international conference on intelligent vehicles, pp 335–340
29. Trucco E, Verri A (1998) Computer vision and applications: a guide for students and practitioners. Prentice Hall, Upper Saddle River, NJ
30. Vassallo RF, Santos-Victor J, Schneebeli HJ (2002) A general approach for egomotion estimation with omnidirectional images. In: IEEE workshop on omnidirectional vision, pp 97–103

Tarak Gandhi received his bachelor of technology degree in computer science and engineering at the Indian Institute of Technology, Bombay. He earned his M.S. and Ph.D. from the Pennsylvania State University in computer science and engineering, specializing in computer vision. He worked at Adept Technology, Inc. on designing algorithms for robotic systems. Currently, he is postdoctoral scholar at the Computer Vision and Robotics Research laboratory at the University of California at San Diego. His interests include computer vision, motion analysis, image processing, robotics, target detection, and pattern recognition. He is working on projects involving intelligent driver assistance, motion-based event detection, traffic flow analysis, and structural health monitoring of bridges.

Mohan Manubhai Trivedi is a professor of electrical and computer engineering at the University of California at San Diego. Trivedi has a broad range of research interests in the intelligent systems, computer vision, intelligent (“smart”) environments, intelligent vehicles and transportation systems, and human-machine interface areas. He established the Computer Vision and Robotics Research Laboratory at UCSD. Currently, Trivedi and his team are pursuing systems-oriented research in distributed video arrays and active vision, omnidirectional vision, human body modeling and movement analysis, face and affect analysis, and intelligent vehicles and interactive public spaces. He serves on the executive committee of the California Institute for Telecommunication and Information Technologies [Cal-(IT)^2] as the leader of the Intelligent Transportation and Telematics Layer at UCSD. He also serves as a charter member of the executive committee of the University of California systemwide Digital Media Innovation Program (DiMI). He serves regularly as a consultant to industry and government agencies in the USA and abroad. Trivedi was editor-in-chief of the Machine Vision and Applications Journal during 1997–2003. He has served on the editorial boards of journals and program committees of several major conferences. He served as chairman of the Robotics Technical Committee of the IEEE Computer Society. He was elected to serve on the administrative committee (BoG) of the IEEE Systems, Man and Cybernetics Society. Trivedi has received the Distinguished Alumnus award from the Utah State University and Pioneer (Technical Activities) and Meritorious Service awards from the IEEE Computer Society. He is a fellow of the International Society for Optical Engineering (SPIE).