Moving Obstacle Detection From a Navigating Robot

Dinesh Nair, *Member, IEEE,* and Jagdishkumar K. Aggarwal, *Fellow, IEEE*

***Abstract*—This paper presents a system that detects unexpected moving obstacles that appear in the path of a navigating robot, and estimates the relative motion of the object with respect to the robot. The system is designed for a robot navigating in a structured environment with a single wide-angle camera. The objective of the system is to detect moving obstacles in order to gradually stop the robot to avoid collision; maneuvering around the obstacle is not considered here. The system has been assembled using pieces of existing vision techniques with a strong emphasis on real-world applications and very fast processing with conventional hardware. The system uses polar mapping to simplify the segmentation of the moving object from the background. The polar mapping is performed with the focus of expansion (FOE) as the center. A vision-based algorithm that uses the vanishing points of segments extracted from a scene in a few three-dimensional (3-D) orientations provides an accurate estimate of the robot orientation. This is used to maintain the motion of the robot along a purely translational path and also used to subtract the effects of any drifts from this path from each image acquired by robot. By doing so, the determination of the FOE is simplified. In the transformed space qualitative estimate of moving obstacles is obtained by detecting the vertical motion of edges extracted in a few specified directions. Relative motion information about the obstacle is then obtained by computing the time_to_impact between the obstacles and robot from the radial component of the optical flow. The system was implemented and tested on an indoor mobile robot at our laboratory. Results from the robot navigating in real environments are presented and analyzed here. The system is able to detect moving obstacles at 100 ms/frame and can be used as a cueing mechanism for moving obstacles.**

***Index Terms*—Edge detection, focus of expansion, navigation, optical flow, polar mapping, robot, time to impact, vanishing points.**

I. Introduction

DETECTING obstacles from a moving platform is one of the key issues to the successful application of mobile robot systems. This problem has attracted a number of computer vision researchers over the years. A wide variety of approaches and algorithms have been proposed to tackle this complex problem, from simple algorithms that detect an obstacle and stop the robot short of the object to avoid collision, to more complex algorithms that estimate the position and dimensions of the obstacle and prepare strategies to maneuver around it. Detecting moving obstacles from a moving robot has received little attention especially for robot systems that use only vision for obstacle avoidance. One of the reasons (and also a key issue) is the inherent difficulty in differentiating between the stationary background and the nonstationary objects, since, from a moving platform, both the background and the object appear to be moving.

The notable algorithms for detecting moving objects from a moving platform using only a vision sensor (camera) use temporal or spatial disparities estimated from the image sequence for detection purposes. These algorithms can be grouped into two distinct classes, namely:

1) methods using optical flow (temporal disparity);
2) methods using qualitative estimates of motion.

Algorithms in the first class ([1]–[5]) first compute the global flow field (optical flow) from the whole image and then use the optical flow information to analyze scenes obtained by a moving observer. A disadvantage with these methods is the difficulty in computing the optical flow with an acceptably low level of noise. The higher level of noise makes the segmentation process highly unreliable in real life applications. This is particularly true for cases where the object covers a large portion of the image or where the motion of the object is sudden and abrupt. However, optical flow is a powerful tool to determine the depth, structure and motion of the object relative to the observer. Some methods have been developed that are solely based on the normal flow rather than the full optical flow field ([6]–[8]). The second class of algorithms ([9]–[13]) use image transformations and qualitative analysis of the motion of scene points to detect and segment the moving obstacles. These methods are typically more effective in the object segmentation process, but additional processing is required to get information about the object's motion. The focus of expansion (FOE) plays an important role in a majority of the algorithms mentioned in both classes. However, determining the FOE in real scenes is a nontrivial problem. Most methods use optical flow vectors obtained from a sequence of frames to determine the FOE. Results obtained for this again depend on the accuracy of the computed motion information. In [14], an approach to compute a fuzzy FOE was presented.

In this paper, we present a system to detect unexpected moving obstacles that appear in the path of a navigating robot and to estimate the relative motion of the object with respect to the robot. The aim of this system is to realize the task of moving obstacle detection from the robot with the aim of a real-time effective implementation. Since the robot does

Manuscript received September 23, 1996; revised January 8, 1998. This work was supported by the Texas Higher Education Coordinating Board, Advanced Research Project 275, and the Army Research Office Contract DAAH 049510494. This paper was recommended for publication by Associate Editor M. Hebert and Editor R. Volz upon evaluation of the reviewers' comments.

D. Nair is with National Instruments, Inc., Austin, TX 78712 USA.

J. K. Aggarwal is with the Computer and Vision Research Center, Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78712-1084 USA.

Publisher Item Identifier S 1042-296X(98)02915-2.

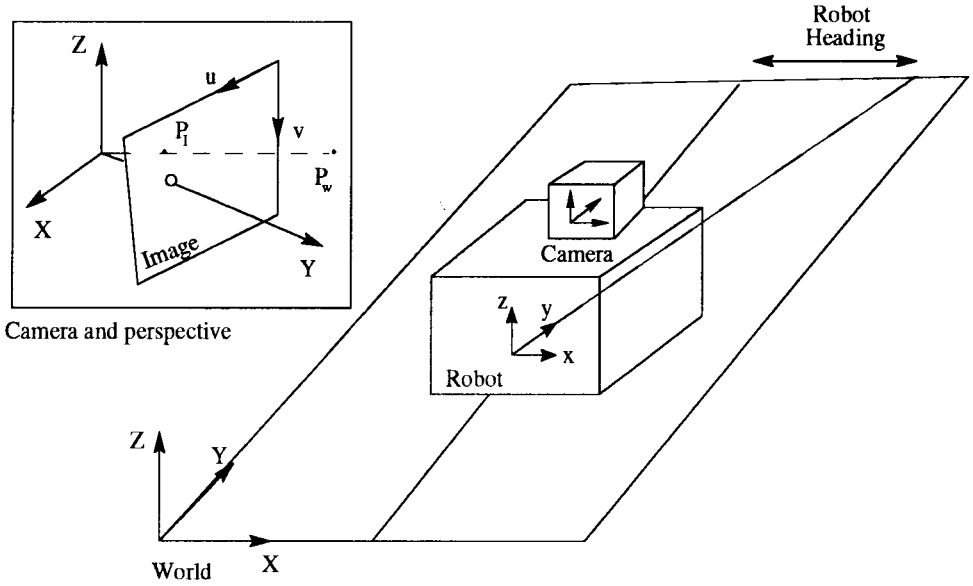


Fig. 1. Coordinate systems.

not maneuver around the obstacle, the need for the complete segmentation of the moving obstacle is not required (which may be time-consuming). The system is proposed for the robot ROBOTEX that navigates in a structured environment with a single wide-angle camera. This approach detects the moving obstacles using qualitative analysis of the scene, similar to the algorithms that belong to the second class of motion detection algorithms mentioned above. The system uses a polar transformation to aid the segmentation of the moving object from the background. The problem of determining the FOE accurately for this detection process is simplified by constraining the motion of the robot (and hence the camera) to almost pure translation. Slight drifts in the robot motion from the translational path is corrected by subtracting the robot's angular motion from each acquired frame in an image sequence. Accurate estimate of the robot's egomotion is obtained from the odometry after correcting it using a vision based algorithm. The vision based algorithm uses vanishing points of lines in three significant three-dimensional (3-D) orientations to accurately estimate the robot's heading, roll, and pitch. After qualitative detection of the moving obstacles, relative motion information of the objects is obtained by computing the time_to_impact between the robot and the obstacles. The time_to_impact is computed from the radial component of the optical flow of the object.

The rest of the paper is organized as follows: Section II describes the navigation environment and the method used to constrain the robot motion to pure translation. The process of subtracting the robot's angular motion from each acquired image is discussed here. Section III describes the process by which a qualitative estimate of the moving objects is obtained. In Section IV the relation between the radial component of the optical flow and the time_to_impact is derived, and the method used to compute the relative motion of the obstacle is described. In Section V the experimental robot, ROBOTEX, is described and implementation results of the proposed system are presented. Conclusions are presented in Section VI.

II. Robot Navigation

The robot navigates in a structured environment (like corridors, hallways, etc.) with a single wide-angle camera. The robot's odometry along with robot heading values computed using a vision based algorithm, are used to restrict the robot's motion close to pure translation. The procedure is described in more detail below.

A. Coordinate Systems

The robot and world coordinate systems used in this paper are as shown in Fig. 1. $\mathbf{W}$ represents the world coordinate system with a vertical z-axis, $\mathbf{R}$ the robot coordinate system, $\mathbf{C}$ the camera coordinate system and $\mathbf{P}$ the image coordinate system (used for the perspective projection on the CCD of the camera). The transformation from $\mathbf{W}$ to $\mathbf{C}$ is given by

$$T_{WC} = T_{WR} T_{RC}. \tag{1}$$

T_{WR}, the homogeneous coordinate transformation matrix from $\mathbf{W}$ to $\mathbf{R}$ is given by

$$T_{WR} = \begin{bmatrix} \cos r & 0 & -\sin r & 0 \\ 0 & 1 & 0 & 0 \\ \sin r & 0 & \cos r & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos p & \sin p & 0 \\ 0 & -\sin p & \cos p & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos h & \sin h & 0 & 0 \\ -\sin h & \cos h & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where, r, p, and h represent the roll, pitch, and heading of the robot. x, y, and z represent the relative translation of the robot with respect to the world and can be accurately obtained from the odometry.

T_{RC}, the coordinate transformation matrix from $\mathbf{R}$ to $\mathbf{C}$, is completely determined through eye/wheel calibration [15]. The calibration procedure is used to determine intrinsic parameters,

including barrel distortion, as well extrinsic parameters (camera–robot relationship). An optical method is used to determine the location of the optical center of the camera in pixel units. The optical center is determined by the position on the camera (image), where a low-power beam shown onto the camera from a carefully aligned laser reflects onto itself. A calibration pattern consisting of a grid of black dots placed every 10 cm is used to determine the focal length and the barrel distortion of the camera. The focal length is computed using points lying close to the optical center, where the distortion is minimal. The focal length is given by $f = lD/L$, where l is the distance in pixels of the imaged points, L is the length in mm between the points on the calibration pattern, and D is the physically measured distance from the location of the CDD of camera to the pattern. An estimate of the barrel distortion is obtained by measuring the two-dimensional (2-D) displacements between the theoretical and actual positions of the pattern points when imaged. Care is taken to ensure that the pattern covers the entire field of view of the camera. The 2-D distortion function is then estimated by linear interpolation between data points. To speed up distortion correction, a lookup table is constructed at calibration time to translate the pixels from the distorted image to the undistorted one.

To calibrate the extrinsic parameters, the translations are physically measured. The roll of the camera is forced to zero by aligning cross hairs digitally superimposed on the image to the horizontal axis of the calibration pattern. The pan and tilt angles of the camera are reduced to zero by adjusting them until the robot can back away from the grid in a straight line, while keeping the same calibration point at the optical center. The resulting T_{RC} transformation is given by

$$T_{RC} = \begin{bmatrix} 1 & 0 & 0 & -x_{rc} \\ 0 & 1 & 0 & -y_{rc} \\ 0 & 0 & 1 & -z_{rc} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where, x_{rc}, y_{rc}, and z_{rc} represent the camera to robot translations.

Finally, the perspective projection from camera to image plane is given by

$$\begin{bmatrix} su \\ sv \\ s \\ 1 \end{bmatrix}_P = \begin{bmatrix} \alpha_u f & u_0 & 0 & 0 \\ 0 & v_0 & -\alpha_v f & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_C \tag{4}$$

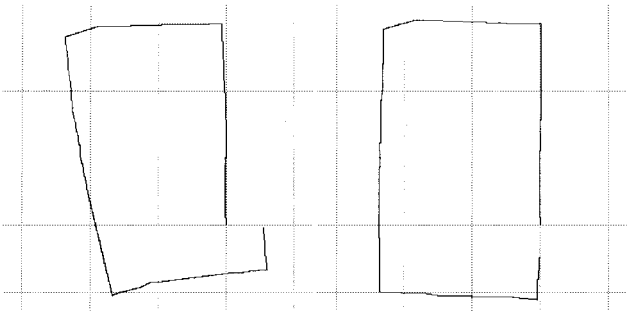
where, u and v represent the coordinates of a point on the image plane (pixels). α_u and α_v are conversion factors (pixels per unit length), u_0 and v_0 are the coordinates of the optical center of the camera (pixels), and f is the focal length (unit length). These parameters are determined through the calibration of the camera as described above. s represents the depth, and therefore cannot be determined from a single image. The *robot heading* is defined as a rotation of the robot about the vertical axis in the world coordinate system.

B. Robot Orientation

Measurements from the robot's odometry provide the relative position and the heading at a given instant. However, since the odometers drift without bounds, these measurements cannot be used alone to maintain the path of the robot. Our robot is equipped with an odometer on its left and right driving wheels. Rotations (changes in the robot's heading) rely on the difference between these odometry readings, and hence are not accurately measured over long periods of time. In practice, they are adequate for estimating the motion between a few images. Therefore, the odometer heading reading is periodically corrected by an algorithm that computes the heading from the vanishing points of the lines extracted in three significant 3-D orientations from an image. Vanishing points have been used for camera calibration [16] and methods to extract vanishing points from indoor and outdoor scenes have been presented in [17] and [18].

(a) (b)

Fig. 2. (a) Typical corridor scene and (b) 2-D lines extracted in three semantically significant 3-D directions. The dot at the center is the location of the vanishing point of the horizontal lines going into the image. This vanishing point is used to estimate the heading and pitch of the robot. The vanishing points of the lines in the other two directions are at infinity.



(a) (b)

Fig. 3. The path of the robot: (a) as estimated by odometry and (b) after correction by vision.

Line segments in three prominent 3-D orientations (one vertical and two horizontal orientations perpendicular to each other), are extracted from an image using a fast line detector [17]. The line detector uses the *a priori* knowledge of the location of the vanishing points to extract line segments. Only one 3-D orientation is considered at a time. The 2-D segments that converge to the corresponding vanishing point are detected and associated with the 3-D orientation. Also, the vanishing points are not explicitly computed. Instead, a vector pointing to the vanishing point is computed for each image pixel using very few operations. Fig. 2 shows a typical image obtained in a man-made corridor and the corresponding vanishing points obtained from the significant 3-D lines in the environment.

The robot's roll, pitch, and heading can be computed from the vanishing points of the extracted 2-D segments. To

(a) (b) (c)

Fig. 4. (a)–(c) Typical corridor scenes used for the experiments reported in this paper.

achieve good precision, these angles should be computed from the vanishing point which varies most with that angle. For example, the heading of the robot (i.e., a rotation around the vertical axis of W) can be obtained by using the vanishing point of the 2-D segments in either of the two horizontal directions. The vanishing point that lies closest to the center of the image is used in this case. Since this vanishing point varies both with the pitch (p) and the heading (h) of the robot, both these angles can be computed from it. By definition, the vanishing point $[u_{vp}v_{vp}]^T$ is given by

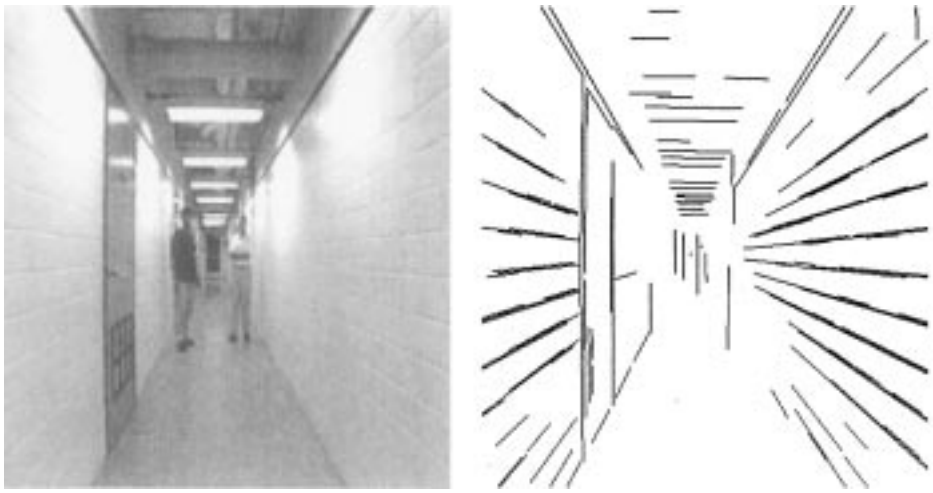
$$u_{vp} = \lim_{s\to\infty} \frac{su}{s}$$
$$v_{vp} = \lim_{s\to\infty} \frac{sv}{s} \tag{5}$$

where, s, u, and v are given by (4). Simplifying by using the fact that the position of the vanishing point is independent of the roll of the camera, the values of the pitch, and heading of the robot can be obtained using (6), as shown at the bottom of the page, where $T = T_{RP}$ the transformation matrix from robot to perspective and $T_{i,j}$ represents the i, j element of matrix T. Similarly the roll of the robot can be computed from the vanishing point that most varies with it [19].

Ideally, to maintain the path of the robot in a given direction (approximately translating) the heading of the robot should be corrected every time it drifts from the path. However, due to physical limitations the robot cannot accurately perform rotations that are below a certain value (10°). Hence, rather than update the robot heading at every time it drifts from the original path, it is done only when the rotation is greater than a certain threshold. In between these rotation updates, the condition of pure translation is maintained by removing the effects of robot rotation from each acquired image. This is done by derotating each image as explained below.

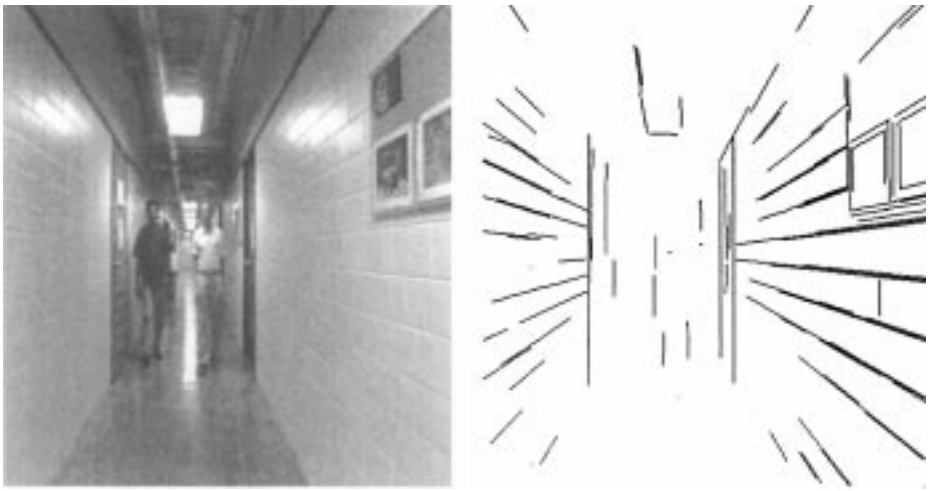
The quality of the results depend on the error associated with the vanishing point extraction and the precision of calibration. To illustrate the effectiveness of the heading computation using the vision algorithm, the path of the robot while navigating along a building corridor is plotted in Fig. 3. Fig. 3(a) shows the path of the robot using only the odometer readings, while Fig. 3(b) with the heading extracted by the vision algorithm combined to the translation given by the odometry. At the beginning of the experiments, the robot was approximately aligned with the center of a hallway. The total distance traveled was 125 m, and 97 images were used to compute the headings. In the robot used for our experiments, the error introduced due to the odometer was systematic and always introduced a slight drift to the left. However, the method described here will work for drifts both to the left and right from the original path. Another experiment was conducted to test the performance of the heading computation in the presence of obstacles (i.e., people and open doors). The robot was programmed to rotate (i.e., change its heading) in increments of 2° up to 10° in three different corridor settings [Fig. 4(a)–(c)]. Images were acquired in the presence of moving objects such as those in Figs. 5(a), 6(a), 7(a), and 8. The algorithm is unaffected by people moving in the field of view unless they occupy a large portion of the field of view as shown in Fig. 8 where the significant 3-D lines in the image cannot be detected. The accuracy in the heading estimation depends on how accurately the lines in the image can be determined. An experiment was conducted to quantify the accuracy in the heading measurement. The robot was initially placed at the center of the corridor facing in a direction parallel to the walls of the corridor (i.e., with a heading of 0°.) The robot was then rotated clockwise by increments of two degrees from 2–10° and the heading values computed by the algorithm described

$$p = \arctan\left[\frac{(T_{1,1}T_{2,0} - T_{2,2}T_{1,0}) * u_{vp} + (T_{2,1}T_{0,0} - T_{0,1}T_{2,0}) * v_{vp} + (T_{0,1}T_{1,0} - T_{1,1}T0,0)}{(T_{1,2}T_{2,0} - T_{2,2}T_{1,0}) * u_{vp} + (T_{2,2}T_{0,0} - T_{0,2}T_{2,0}) * v_{vp} + (T_{0,2}T_{1,0} - T_{1,2}T_{0,0})}\right]$$
$$h = \arctan\left[\frac{T_{0,1}\cos p - T_{0,2}\sin p - (T_{2,1}\cos p - T_{2,2}\sin p) * u_{vp}}{T_{2,0}u_{vp} - T_{0,0}}\right] \tag{6}$$



(a) (b)

Fig. 5. Estimating heading accuracy as computed by the vision algorithm for: (a) scene in corridor 1 and (b) the lines extracted from the image and the heading estimate. The dot at center of the image shows the current heading and the dot on its left gives the desired heading (or vanishing point).



(a) (b)

Fig. 6. Estimating heading accuracy as computed by the vision algorithm for: (a) scene in corridor 2 and (b) the lines extracted from the image and the heading estimate.

above was recorded. Table I gives the accuracy in the heading measurements for these experiments. The average error in the heading estimation is about 2.3%.

C. Subtracting Angular Movements of the Robot From an Image

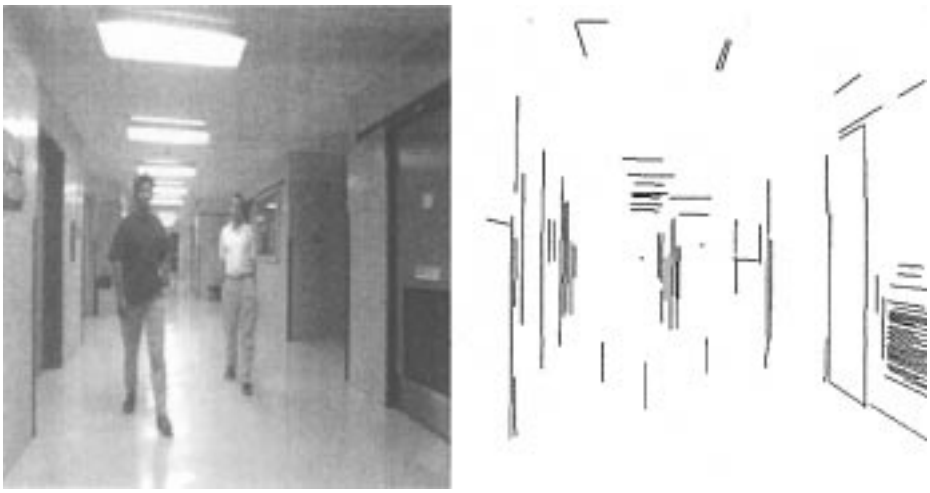
An accurate estimate of the robot heading, pitch, and roll at any instant of time is either obtained from its odometry or computed using vanishing points from an image. In practice, however, it was noticed that roll and pitch of the robot time vary very little and are almost constant. This is true when the robot navigates on even floors. Once the angular motion of the robot is known, it can be subtracted from each image using a simple transformation as derived below.

The rotation of the projection onto the image plane of a point in the world plane due to the rotation of the robot (i.e., heading) can be obtained through the following steps. First, determine the location of the world point, $\overline{\mathbf{X}} \equiv (X, Y, Z)$, in the image plane assuming that the robot has undergone a rotation. Let this location on the image plane be (u, v). Then, (u, v) is given by

$$\begin{bmatrix} su \\ sv \\ s \\ 1 \end{bmatrix} = T_{CP} \cdot T_{RC} \cdot T_{WR} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_W \tag{7}$$

$$= T_{CP} \cdot T_{RC} \cdot T_h \cdot T_t \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_W \tag{8}$$

where, T_{CP}, T_{RC}, and T_{WR} are the transformation matrices from the world coordinate system to the image plane defined by (1)–(4). Next, determine the location of the same world point $\overline{\mathbf{X}}$ on the image plane if the robot had not undergone a rotation. Let this location on the image plane be denoted by



(a) (b)

Fig. 7. Estimating heading accuracy as computed by the vision algorithm for: (a) scene in corridor 3 and (b) the lines extracted from the image and the heading estimate.

TABLE I
ACCURACY ESTIMATION OF HEADING ANGLE COMPUTATION

Angle Range	Corridor 1		Corridor 2		Corridor 3	
	Estimated	Accuracy	Estimated	Accuracy	Estimated	Accuracy
2^o	2.008	0.4%	2.012	0.6%	2.2	1.1%
4^o	4.072	1.8%	4.085	2.12%	4.12	2.3%
6^o	6.117	1.95%	6.15	2.25%	6.145	2.42%
8^o	8.231	2.88%	8.242	3.03%	8.104	3.13%
10^o	10.315	3.15%	10.325	3.25%	10.38	3.8%

Fig. 8. Example of a scene where the heading accuracy degrades due to significant occlusion of lines in the scene by the obstacles.

$(\acute{u}, \acute{v})$. Then, $(\acute{u}, \acute{v})$ is given by

$$\begin{bmatrix} \acute{s}\acute{u} \\ \acute{s}\acute{v} \\ \acute{s} \\ 1 \end{bmatrix} = T_{CP} \cdot T_{RC} \cdot T_t \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_W . \tag{9}$$

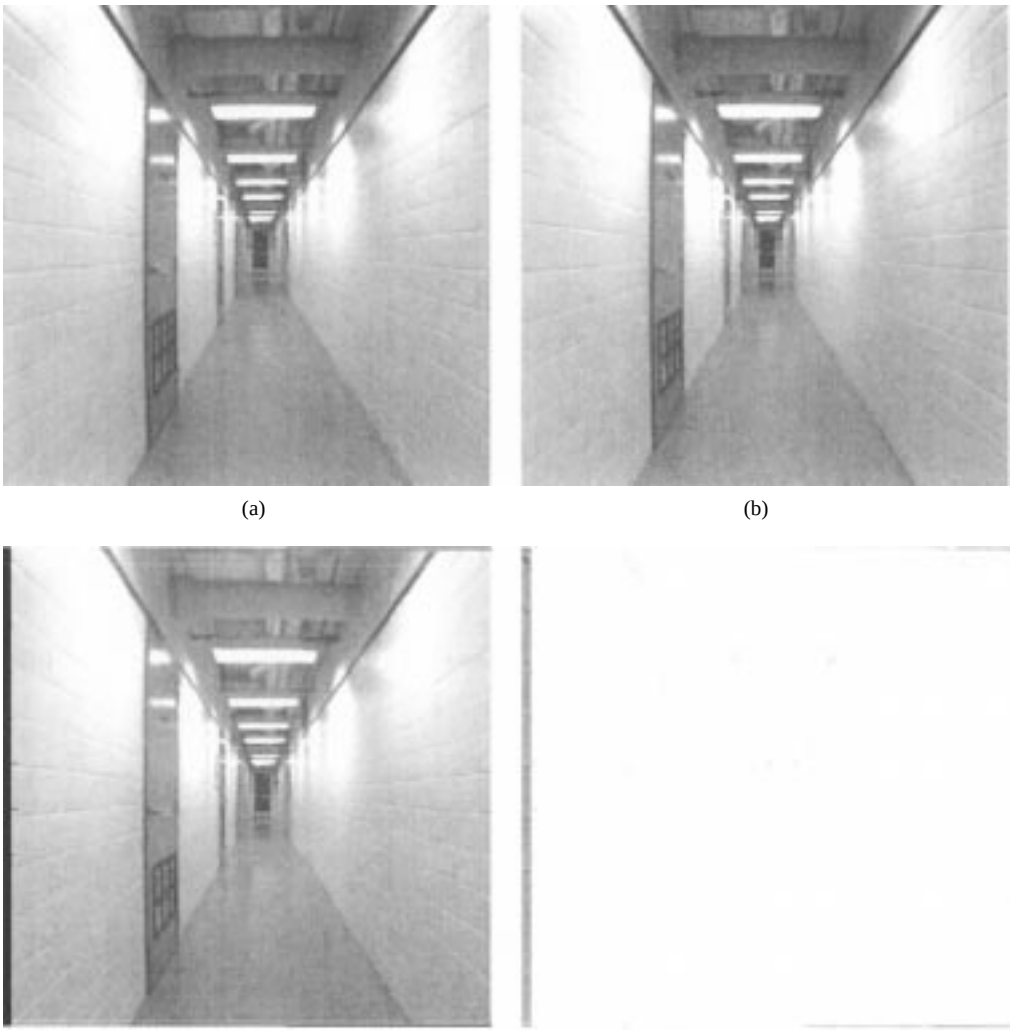
Using (8) and (9), we need to arrive at a transformation that maps (u, v) to $(\acute{u}, \acute{v})$ for all values of u and v in an image. The resulting image, where the pixel locations are given by $(\acute{u}, \acute{v})$ represents the image of a scene from which the rotation of the robot has been removed. Simplifying the above equations using (1)–(4), the angular rotation of the robot can be removed from each point in an image using the following transformation:

$$\begin{bmatrix} \acute{s}\acute{u} \\ \acute{s}\acute{v} \\ \acute{s} \\ 1 \end{bmatrix} = T_{CP} \cdot T_{RC} \cdot T_h^{-1} \cdot T_{RC}^{-1} \cdot T_{CP}^{-1} \cdot \begin{bmatrix} su \\ sv \\ s \\ 1 \end{bmatrix} \tag{10}$$

where T_h^{-1} is the inverse of the rotation matrix associated with the heading of the robot [given in (2)]. Since this matrix is orthonormal, its inverse is obtained as the transpose of the corresponding rotation matrices. T_{CP}^{-1} represents the inverse of the camera to perspective transformation matrix T_{CP} (4) and T_{RC}^{-1} represents the inverse of the robot to camera transformation matrix, and are given by the relations

$$T_{CP}^{-1} = \begin{bmatrix} \frac{1}{\alpha_u f} & 0 & \frac{-u_0}{\alpha_u f} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & \frac{-1}{\alpha_v f} & \frac{v_0}{\alpha_v f} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{RC}^{-1} = \begin{bmatrix} 1 & 0 & 0 & x_c \\ 0 & 1 & 0 & y_c \\ 0 & 0 & 1 & z_c \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{11}$$



(a) (b) (c) (d)

Fig. 9. (a) Image with heading of robot = 0, (b) robot heading = 3°, (c) image in (b) derotated to compensate for heading, and (d) error image: difference between the image in (a) and the image in (c).

Expanding the transformation matrices in (10) and simplifying (assuming that $x_c = 0$ and $y_c = 0$), we arrive at

$$\acute{u} = \frac{(u-u_0)\left[\dfrac{u_0 s(h)}{\alpha_u f} + ch\right] - u_0 c(h) - \alpha_u f s(h)}{(u-u_0)\dfrac{s(h)}{\alpha_u f} + ch}$$

$$\acute{v} = \frac{(u-u_0)\dfrac{v_0 s(h)}{\alpha_u f} + (v-v_0)}{(u-u_0)\dfrac{s(h)}{\alpha_u f} + ch} \tag{12}$$

where $s(\theta) \equiv \sin(\theta)$ and $c(\theta) \equiv \cos(\theta)$.

Fig. 9(a) shows image where the robot is heading in the direction of the FOE, Fig. 9(b) where its heading is 3° with respect to the FOE. Fig. 9(c) shows the image after the image Fig. 9(b) is derotated to compensate for the heading, and Fig. 9(d) shows the error between the images in Fig. 9(a) and (c).

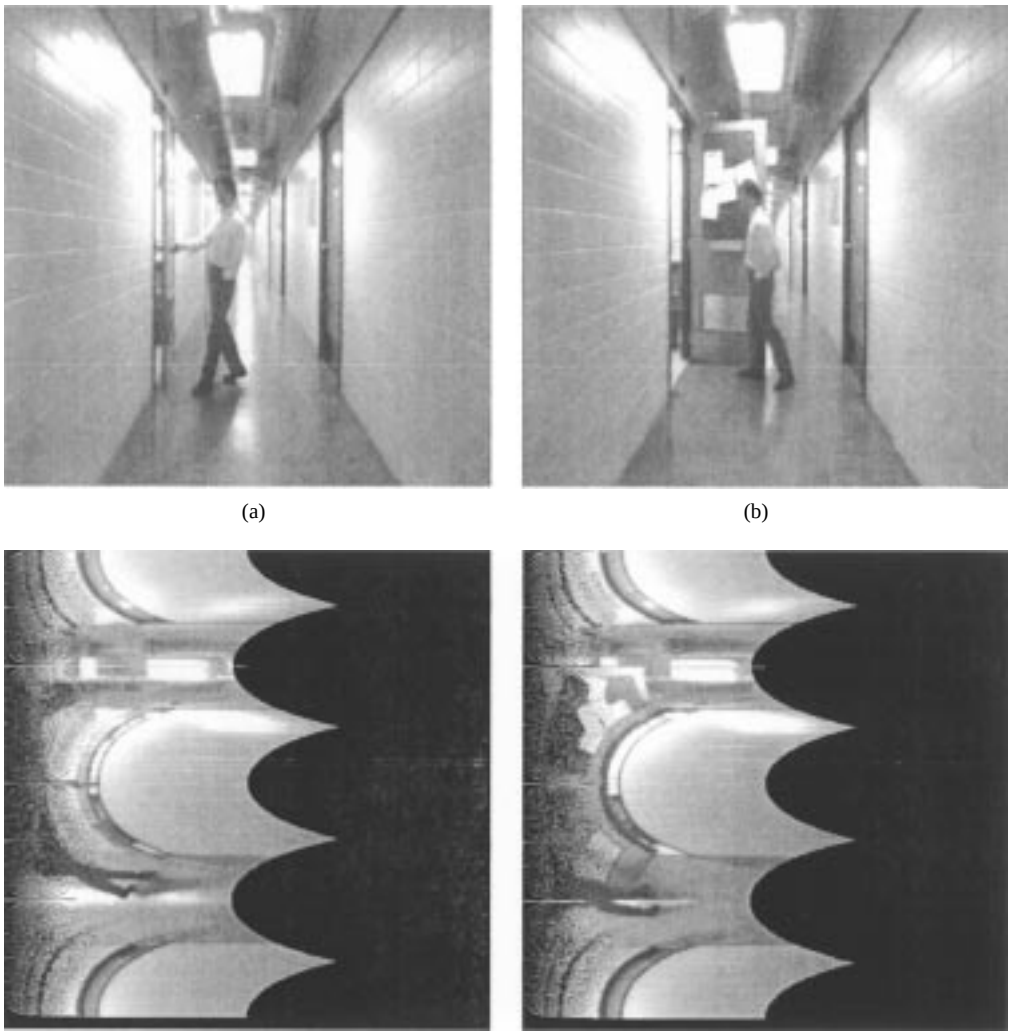
Hence, by correcting the path of the robot at regular intervals and by subtracting the angular movements of the robot from each acquired image, almost pure translation is maintained.

III. Qualitative Detection of Moving Objects

The first step in detecting the moving obstacles in the robot's path is done in a qualitative manner because, in practice, estimation using optical flow techniques is noise prone and usually fails when motions are abrupt and large. In a typical corridor, common obstacles are in the form of people walking or doors that are flung open suddenly. A qualitative estimate of such motion can provide sufficient information to control the robot's action. Such an estimate is obtained by using a polar mapping to segment the moving object(s) from the background. The various stages in this segmentation process are described next.

A. Polar Mapping of Images

To segment moving objects from the background, each image acquired by the robot is transformed from Cartesian coordinates to polar coordinates, using a polar mapping (PM) that transforms the image to a polar coordinate system with the FOE as the center. The FOE is that distant point in the environment toward which the moving platform is translating,



(a) (b) (c) (d)

Fig. 10. (a), (b) Two images from a typical sequence in a hallway. The image in (b) has been derotated to compensate for robot heading. (c), (d) Polar mapping of the images, represented with the axes ρ and η in Cartesian coordinates.

whereas all the stationary objects in the environment move radially away from the FOE. The image is transformed to polar coordinates using

$$\rho = \sqrt{(x - x_{\mathrm{FOE}})^2 + (y - y_{\mathrm{FOE}})^2}$$

and

$$\eta = angle(x - x_{\mathrm{FOE}},\, y - y_{\mathrm{FOE}}) \tag{13}$$

where ρ is the radial distance from the FOE to the rectangular image coordinate (x, y), and η represents the angle (from 0 to 2π) subtended by the rectangular image coordinates (x, y), the FOE, and the rectangular image coordinate (1, 0).

The advantages of using the polar and the log-polar [or complex logarithmic mapping (CLM)] for detecting and segmenting moving objects from moving platforms have been shown in [10] and [11]. For the complex logarithmic mapping, the transformation is similar to the one shown above, except that the logarithm of the radius is used. Although both of these transformations give almost identical results for detecting and segmenting the moving objects in a qualitative manner, quantization errors degrade the performance of the CLM approach when trying to get a quantitative estimate of the object (for example when trying to compute the optical flow). Hence the polar mapping was used for all experiments described here.

If this transformed image is represented with the radial and angular axes as the Cartesian coordinates (as shown in Fig. 10), then the motion of the stationary objects will be in the horizontal direction [i.e., increasing radius (ρ)]. Objects that are moving with respect to the moving observer will have motion in an angular direction in this transformed space. Thus, by detecting this angular motion, the moving objects can be segmented from the background.

To perform the above mapping, the FOE must be located accurately. Determining the FOE automatically is a complex problem in itself [14]. The problem of accurately determining the FOE is circumvented here by constraining the robot motion to be purely translation. This is achieved by regularly updating the heading of the robot using the vision based algorithm described in Section II. In the intervals between the updates, the robot's angular movements are delineated from each acquired image using the transformation given by (10). Using the above methods, the robot is constrained to translate

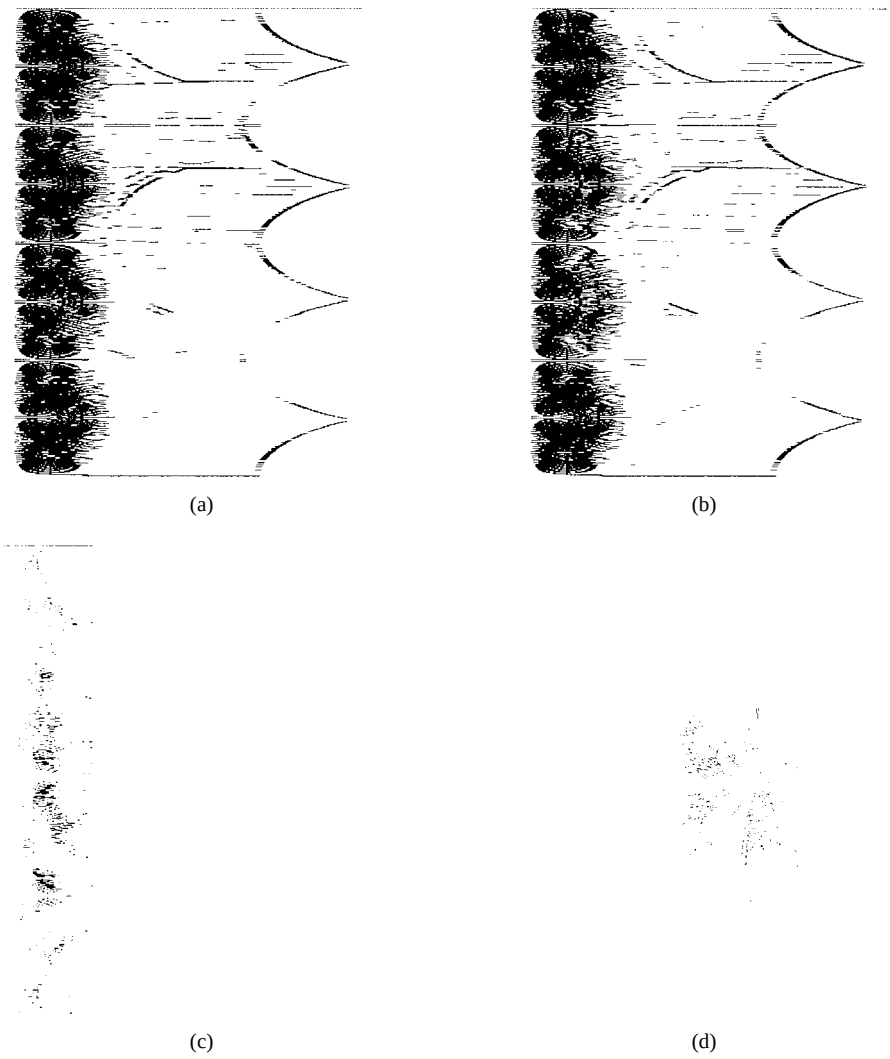


Fig. 11. (a), (b) The edges detected in the images Fig. 10(c) and (d), (c) detected motion obtained by (a)*(a)–(a)*(b), and (d) the detected motion transformed back into rectangular coordinates.

toward a distant point whose perspective projection onto each acquired 2-D image is given by the location of the optical center of the camera on the image plane.

B. Detecting Vertical Motion

After transforming the image to polar coordinates, the problem of detecting a moving object is reduced to finding vertical motion along an angular axis in a sequence of transformed images. A qualitative measure of the motion is obtained by detecting the vertical motion of edges (horizontal and angular) present in the transformed image. The method used in here is similar to that used in [10], which used horizontal edges in the transformed images to determine vertical motion in consecutive images. However, better detection of the moving object is obtained by combining the vertical motion information of edges oriented in different directions with that obtained from the horizontal edges. In this case, edges oriented at $\pm 45^{\circ}$ were used. Let I_i and I_{i+1} be the polar transformed images of the ith and $(i+1)$th images in a typical image sequence. A qualitative estimate of the motion of the object in these images is obtained as follows. First, horizontal and angular edges in the transformed images are enhanced. Let $I_{(i)sobel}$ and $I_{(i+1)sobel}$ represent the resultant images obtained by convolving I_i and I_{i+1} with Kirsh's [20, pp. 79–80] kernels, respectively. Then, the image I_{mot} obtained as

$$I_{mot} = I^2_{(i)sobel} - I_{(i)sobel} \cdot I_{(i+1)sobel}, \tag{14}$$

is a map of all horizontal and angular edges that moved vertically in image $I_{(i+1)sobel}$ since the earlier frame $[I_{(i)sobel}]$. Some edges that have moved horizontally may be present in this resultant image, but they are usually very small pieces and can be removed by suitable thresholding. This image contains the detected motion.

Figs. 10 and 11 goes through the entire qualitative motion detection process for two frames in a sequence. Fig. 10(a) and (b) show two images from a typical sequence. Fig. 10(c) and (d) show the polar mapped representations of the original images. Fig. 11(a) and (b) show the edges detected in the polar mapped images. Fig. 11(c) shows the resulting image obtaining after subtracting the product of the images in Fig. 11(a) and (b) from the image in Fig. 11(a). This image contains

the detected motion. Fig. 11(d) shows the detected motion transformed back to the rectangular frame.

IV. Relative Motion of Obstacles

After obtaining a qualitative measure of the motion of the moving object in front of the navigating robot, the presence of a moving object can be easily detected using the procedure outlined above. This information itself can be used to modify the navigation parameters of the robot. In the simplest case, the robot can be halted at the first instance of detected motion. However, it is important to know the relative motion and location of the objects with respect to the robot for navigation purposes. In practice, for navigation in corridors (like in the experiments considered here), the knowledge of the relative motion of the object is more important that its position. This is due to the lack of space for the robot to maneuver around moving objects in narrow corridors. Hence, we consider only determining the relative motion of the object. A method for determining the region in the image occupied by the object (region of interest) is described in [21].

A. Obtaining the Time_to_impact

The relative motion of moving object(s) is obtained by computing the time_to_impact between the robot and the object(s). Relative motion is computed (whenever possible, as explained later) at all regions in an image where qualitative motion was detected. No attempt is made to identify the different moving objects in a scene. The smallest time_to_impact value obtained is used for deciding the next navigation move. Time_to_impact is obtained from the radial component of the optical flow computed from a sequence of images. There are an abundance of equations in the polar and log polar transformed domain to compute the time_to_collision from the optical flow [22], [23]. As mentioned earlier, quantization errors, although present in both the transformed spaces, give larger (unacceptable) errors when the log polar transformed domain is used. Given a fairly dense (five to ten images/s) sequence of images, an accurate estimate of the optical flow can be obtained using any one of the standard techniques [24].

Assuming general motion of the camera as both rotational and translational, the velocity of the image plane along the radial and angular coordinates (for small angular rotations) is

$$\dot{\rho} = \frac{1}{Y}\left[\rho V_z - F(V_x \cos\eta V_z \sin\eta)\right] + \left(\frac{\rho_2}{F} + F\right)(\phi \sin\eta - \theta\cos\eta) \quad (15)$$

$$\dot{\eta} = \frac{F}{\rho}\left[\left(\frac{V_x}{Y} + \theta\right)\sin\eta + \left(\phi - \frac{V_z}{Y}\right)\cos\eta\right] - \psi \quad (16)$$

where, (V_x, V_y, V_z) is the relative translational velocity, (ϕ, θ, ψ) are the components of rotational motion referred to the camera coordinate axes as shown Fig. 1, Y is the distance of the world point from the image plane, ρ is the radial coordinate of the world point on the image plane and η is the angular coordinate of the world point in the image plane. The partial derivatives of $\dot{\rho}$ with respect to ρ are

$$\frac{\partial\dot{\rho}}{\partial\rho} = \frac{V_y}{y} + 2\frac{\rho}{F}(\phi\sin\eta - \theta\cos\eta) \quad (17)$$

$$\frac{\partial^2\dot{\rho}}{\partial^2\rho} = \frac{2}{F}(\phi\sin\eta - \theta\cos\eta). \quad (18)$$

From the above two equations, we obtain (as shown in [22])

$$\frac{Y}{V_y} = \left[\frac{\partial\dot{\rho}}{\partial\rho} - \rho\frac{\partial^2\dot{\rho}}{\partial^2\rho}\right]^{-1} \quad (19)$$

where, Y/V_y represents the time_to_impact. This equations shows that the polar mapping allows us to compute the time_to_impact from the radial component of the optical flow, i.e., by analyzing the rate of expansion of the objects in the scene. The above equations that are used to estimate the time_to_impact are valid only when the FOE is at the polar or log-polar center, which is the case in the application described in this paper.

For the experiments reported in this paper the optical flow was computed using the feature based estimation method reported in [25]. This approach is based on the extraction and interframe match of features. Features are extracted using the monotonocity operator, which classifies each pixel in the image into one of the classes 0 to 8 according to the number of gray values in the pixel's immediate neighborhood, which are less than the gray value of the pixel. It is seen that pixels in each class tend to form blobs. The optical flow is computed by component labeling and matching components of the same class over the two images. Component labeling is achieved through blob coloring and determining the area and centroid of each blob. The area, centroid and feature class information are used to match components over the two images and, hence, to compute the optical flow.

V. Experimental Results

The system was implemented and tested on an indigenously fabricated autonomous mobile robot (Robotex) [26]. RoboTex is a 1.5 m tall, tetherless mobile robot, weighing about 150 kg (see Fig. 12). The robot subsystems are described in detail below. They are comprised of:

1) TRC Labmate base and rigid metal frame to support the equipment;
2) fast, on-board UNIX workstation to digitize video images and control the robot;
3) camera and digitizer;
4) I/O system;
5) power supplies, which enable completely autonomous operation.

The Labmate base can carry 90 kg of equipment at speeds of up to 1 m/s, and accelerations of 10 cm s^{-2}. We use it at 40 cm/s and 5 cm s^{-2} to avoid wheel slippage and remove motion blur. The right and left driving wheels are mounted on a suspension for a good floor contact. Passive casters in each corner ensure stability. The Labmate controller processes measurements from the right and left odometers to update the 2-D position and heading of the robot. We found that, provided the accelerations were reasonable, the odometric readings were

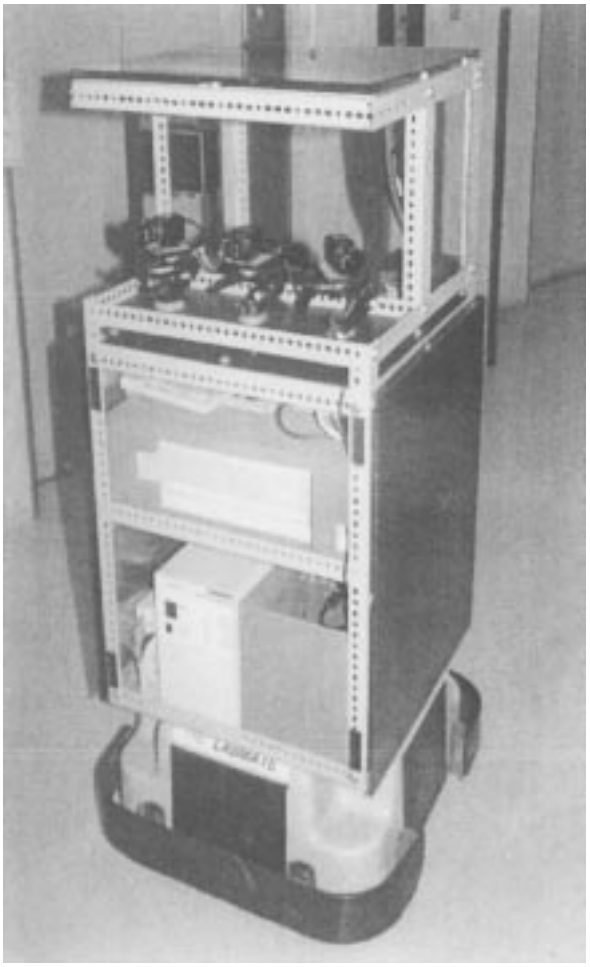


Fig. 12. Robotex: the experimental robot at CVRC.

TABLE II
TIME_TO_IMPACT ACCURACY ESTIMATION FOR (a) SUBJECT 1 AND (b) SUBJECT 2. THE DISTANCE COLUMN DENOTES THE DISTANCE OF THE SUBJECT FROM THE ROBOT

Distance (cm)	True TTI (sec)	Estimated TTI (sec)	% Error
1398	9.04	10.26	13.47
1296	8.38	9.27	10.62
1194	7.72	8.48	9.84
1092	7.06	7.907	10.12
990	6.40	6.94	8.59
888	5.74	6.20	8.10
786	4.42	4.71	6.58
684	3.76	4.02	6.45
582	3.10	3.30	6.96
480	2.44	2.27	6.4
378	1.78	1.893	6.3
276	1.12	1.193	6.5

(a)

Distance (cm)	True TTI (sec)	Estimated TTI (sec)	% Error
1567.5	12.56	10.27	18.2
1485	11.9	13.58	14.12
1402.5	11.24	12.5	11.2
1320	10.61	11.87	11.87
1237.5	9.95	10.81	8.61
1155	9.29	8.43	9.26
1072.5	8.63	7.85	9.03
990.0	7.97	8.63	8.28
907.5	7.31	7.89	7.93
825.0	6.65	7.08	6.47
742.5	5.99	6.37	6.34
660.0	5.33	5.63	5.63
577.5	4.67	4.39	5.99
495.0	4.01	3.79	5.48
412.5	3.35	3.15	5.9
330.0	2.69	2.52	6.3
247.5	2.03	2.15	5.91
165.0	1.37	1.45	5.83

(b)

reliable. A rigid metal frame is bolted onto the Labmate base to carry all the equipment. Rigidity is very important since the transformation between the coordinate systems of the robot and the camera must be calibrated precisely. The main computer on the robot is an HP-735 UNIX workstation, rated at 124 MIPS and 40 MFLOPS with a 99-MHz CPU The robot has a camera (Panasonic WV-CD 50) equipped with a wide-angle lens (6 mm Computar TV lens, for a 2/3 in CCD camera). Barrel distortion introduced by the wide-angle lens is corrected using a bilinear interpolation procedure [15]. The monochrome or RGB video signal from the camera is digitized by a Chorus PC-EYE frame grabber, providing either 8 b for monochrome or 15 b for color. The image size is 512×484.

Extensive tests were conducted to test the system. The effectiveness of the qualitative estimate of the motion was tested on numerous runs in the typical building corridors. The obstacles in these tests were in the form of moving humans and opening doors. Fig. 13(a) and (b) show the start and end frames of a typical sequence. In this sequence there are two subjects walking toward the moving robot. The robot is moving at a speed of 40 cm/s while the subject closest to the robot is walking at a speed of 114.5 cm/s and the second subject is walking at a speed of 85 cm/s. The initial distance of the closer subject from the robot is 1500 cm while that of the other subject is 1650 cm. Three consecutive frames from the same sequence are shown in Fig. 13(c)–(e). Fig. 13(f) and (g) show the areas (transformed back to the Cartesian coordinates) where the motion of the moving objects have been detected in the second and third frames. The true time_to_impact values and the estimated time_to_impact values are given in Table II(a) for subject 1 and Table II(b) for subject 2. The time_to_impact values at intervals of 0.66 s are presented in the table. The time_to_impact values for each subject are obtained by clustering the time_to_impact values into either one or two classes using the method described in and taking the average time_to_impact within each cluster. From these tables we see that the time_to_impact values are fairly consistent across normal human walking speeds. Table II also shows the true time_to_impact and the estimated time_to_impact at varying distances of the moving objects from the robot. The distances between the robot and the subject were obtained by using the blueprints of the floor plan of the corridor the experiment was conducted in. These results show that the time_to_impact values become reliable when the subject is less than 900 cm from the robot and the error rate is about 6%.

For navigation purposes, we make the assumption that the average walking speed of a human is known and this information is used to estimate an approximate distance between the robot and the human. In many cases, although a qualitative estimate of the motion is obtained, the optical flow cannot be computed accurately. These cases arise when the object motion is very abrupt and information about the object is present in only one of the images. In such cases, time_to_impact values are not very reliable and the qualitative estimate of motion provides the only reliable source of motion information.

An indication of the real-time performance of this system can be obtained by computing the time taken from acquiring an image to computing the time_to_impact. The following steps occur during this time:

1) acquire an image from camera sensor;
2) acquire orientation of robot;
3) derotate image;
4) perform polar mapping of the image;
5) extract horizontal and angular edges from the image;

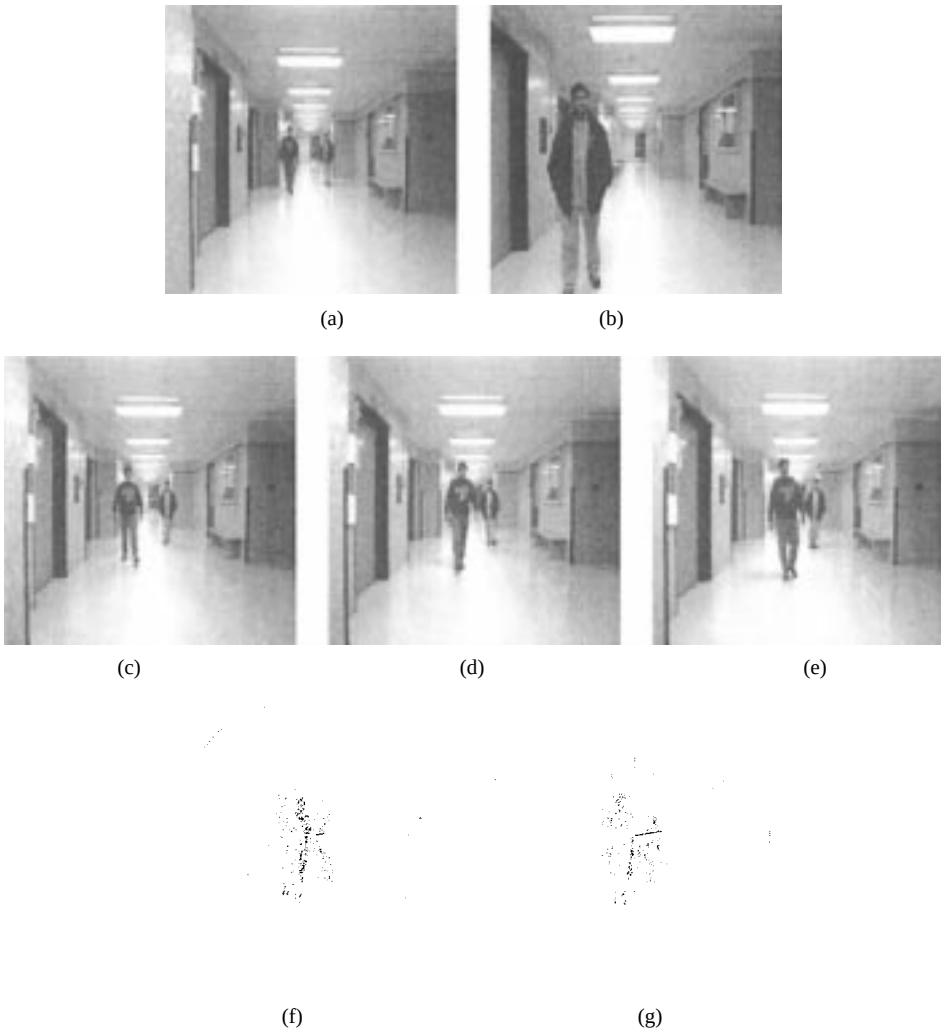


Fig. 13. (a) and (b) Start and end frames of the sequence used to verify accuracy of time_to_impact measurements. Subject 1 is the person closer to the robot. (c)–(e) Three successive images from the sequence. (f) and (g) Show areas in the image where motion has been detected.

6) detect qualitative motion;
7) compute optical flow;
8) determine time_to_impact.

On a PA-RISC based HP-735 workstation, running at 99 MHz, the time taken to perform the above, when Step 2 is obtained directly from the odometry of the robot, is 100 ms. An additional 7 ms is required when the orientation of the robot is acquired from the vanishing points of significant lines in the image.

VI. Conclusion

In this paper, we have presented a system to detect unexpected moving obstacles that appear in the path of a moving robot. The system has been proposed for navigation in a man-made environment such as corridors and is designed to detect moving objects and determine the time_to_impact of the robot to the object. The emphasis has been in forming a system running nearly in real-time and in real situations by integrating an appropriate choice of vision techniques. The detection of the moving object is simplified with the use of polar mapping using the FOE as the center. The problem of determining the FOE accurately is simplified by constraining the motion of the robot to almost pure translation. This is done by using the robot motion readings obtained from the odometry corrected using a vision-based algorithm. Additional precision is obtained by removing the effects of small drifts in the robot motion from its original path using a derotation transformation from each new frame acquired by the robot camera. The polar transformation reduces the problem of detecting the moving object to detecting vertical motion in the transformed space. The relative time_to_impact of the robot is computed (whenever possible) from the radial component of the optical flow in the image plane. Segmentation of the image prior to computation of the optical flow reduces the uncertainties associated with optical flow computations over the whole image and reduces the computation complexity. However, because the proposed system uses fewer areas in the image over which the optical flow and hence the time_to_impact is computed and the smallest time_to_impact measurement is used to alert the robot, the system is sometimes prone to isolated erroneous measurements.

The system was implemented and tested on a mobile robot developed in our lab. The system is able to detect moving obstacles at 100 ms/frame and therefore can be used as a

cueing mechanism for moving obstacles in structured indoor environments (mainly corridors). To our knowledge, there are no documented experimental demonstrations of systems that detect moving obstacles from a moving platform with comparable performance as the system presented in this paper.

REFERENCES

[1] Q. Zhu, "Structural pyramids for representing and locating moving obstacles in visual guidance of navigation," in *IEEE Comput. Society Conf. Comput. Vis. Pattern Recog.*, Ann Arbor, MI, 1988, pp. 832–837.

[2] J. Lee and C. Lin, "A novel approach to real-time motion detection," in *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, Ann Arbor, MI, 1988, pp. 730–735.

[3] W. Thompson and T. Pong, "Detecting moving objects," *Int. J. Comput. Vis.*, vol. 4, pp. 39–57, 1988.

[4] W. Enkelmann, "Obstacle detection by evaluation of optical flow fields from image sequence," *Image Vis. Comput.*, no. 9, pp. 160–168, 1991.

[5] Q. Zheng and R. Chellappa, "Motion detection in image sequences acquired from a moving platform," in *Int. Conf. Acoust., Speech Sig. Process*, 1993, pp. V.201–V.204.

[6] J. Odobez and P. Bouthemy, "Detection of multiple moving objects using multiscale mrf with camera motion compensation," in *1st IEEE Int. Conf. Image Process*, Austin, TX, 1994, vol. 2, pp. 257–261.

[7] B. K. P. Horn and J. Weldon, "Computationally efficient methods for recovering translational motion," in *Int. Conf. Comput. Vis.*, 1987, pp. 2–11.

[8] C. Fermuller, "Global 3-D motion estimation," in *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, New York, 1993, pp. 415–421.

[9] B. Bhanu and W. Burger, "Qualitative motion detection and tracking of targets from a mobile platform," in *DARPA Image Understanding Workshop*, Cambridge, MA, 1988, pp. 289–313.

[10] J. Frazier and R. Nevatia, "Detecting moving objects from a moving platform," in *1992 IEEE Int. Conf. Robot. Automat.*, Nice, France, 1992, pp. 1627–1633.

[11] R. Jain and O. Brien, "Ego-motion complex logarithmic mapping," in *SPIE Int. Soc. Optic. Eng.*, 1984, vol. 521, pp. 16–23.

[12] S. Badal, R. Draper, and A. Hanson, "A practical obstacle detection and avoidance system," in *IEEE Workshop Appl. Comput. Vis.*, Saratoga, FL, 1994, pp. 97–104.

[13] R. Jain, S. L. Bartlett, and N. O'Brian, "Motion stereo using ego-motion complex logarithmic mapping," *Pattern Anal. Machine Intell.*, vol. 9, no. 3, pp. 356–369, 1987.

[14] B. Bhanu and W. Burger, "On computing a 'fuzzy' focus of expansion for autonomous navigation," in *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 1989, pp. 563–568.

[15] X. Lebègue and J. K. Aggarwal, "A mobile robot for visual measurements in architectural applications," in *Proc. IAPR Workshop Machine Vis. Appl.*, Tokyo, Japan, Dec. 1992, pp. 195–198.

[16] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *Int. J. Comput. Vis.*, vol. 4, pp. 127–139, Mar. 1990.

[17] X. Lebègue and J. K. Aggarwal, "Detecting 3-D parallel lines for perceptual organization," in *Proc. 2nd Euro. Conf. Comput. Vis.*, Santa Margherita Ligure, Italy, May 1992, pp. 720–724.

[18] M. Straforini, C. Coelho, and M. Campani, "Extraction of vanishing points from images of indoor and outdoor scenes," *Image Vis. Comput. J.*, vol. 11, no. 2, pp. 91–99, 1993.

[19] X. Lebègue and J. K. Aggarwal, "Significant line segments for an indoor mobile robot," *IEEE Trans. Robot. Automat.*, vol. 9, pp. 801–815, Dec. 1993.

[20] D. Ballard and C. Brown, *Computer Vision.* Englewood Cliffs, NJ: Prentice-Hall, 1982.

[21] D. Nair and J. K. Aggarwal, "Detecting unexpected obstacles that appear in the path of a navigating robtot," in *Proc. 1st IEEE Int. Conf. Image Process.*, Austin, TX, Nov. 1994, vol. 2, pp. 311–315.

[22] M. Tistarelli and G. Sandini, "Direct estimation of time_to_impact from optical flow," in *Proc. IEEE Workshop on Visual Motion*, Princeton, NJ, Oct. 7–9, 1991, pp. 226–233.

[23] _____, "On the advantage of polar and log-polar mapping for direct estimation of time-to-impact from optical flow," *Pattern Anal. Machine Intell.*, vol. 15, no. 4, pp. 401–410, 1993.

[24] J. Barron and D. Fleet, "Performance of optical flow techniques," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Champaign, IL, 1994, pp. 236–242.

[25] W. Enkelmann, R. Kories, H. H. Nagel, and G. Zimmermann, "An experimental investigation of estimation approaches for optical flow fields," in *Motion Understanding: Robot and Human Vision*, W. Martin and J. K. Aggarwal, Eds. Norwell, MA: Kluwer, 1996, ch. 6, pp. 189–226.

[26] X. Lebègue and J. K. Aggarwal, "Robotex: An autonomous mobile robot for precise surveying," in *Proc. Int. Conf. Intell. Auton. Syst.*, Pittsburgh, PA, Feb. 1993, pp. 460–469.

Dinesh Nair (S'93–M'97) received the B.S. degree from the University of Bombay, India, in 1990, the M.S. degree from the University of Houston, Houston, TX, in 1992, and the Ph.D. degree in electrical and computer engineering from The University of Texas at Austin, in 1996.

During the course of his Ph.D., he worked at the Computer and Vision Research Center at The University of Texas at Austin. He is currently with the Analysis Research and Development group at National Instruments, Austin, and heads the machine vision software group. His research interests include robotics, computer vision, automatic target recognition, statistical image processing, industrial machine vision, and neural networks.

Jagdishkumar K. Aggarwal (F'76) has served on the faculty of The University of Texas at Austin College of Engineering since 1964 and is currently the Cullen Professor of Electrical and Computer Engineering and Director of the Computer and Vision Research Center. His research interests include computer vision, parallel processing of images, and pattern recognition. He is author or editor of seven books and 31 book chapters and author of more than 170 journal papers, as well as numerous proceedings papers and technical reports.

Dr. Aggarwal received the Senior Research Award of the American Society of Engineering Education in 1992, and was recently named as the recipient of the 1996 Technical Achievement Award of the IEEE Computer Society. He has served as Chairman of the IEEE Computer Society Technical Committee on Pattern Analysis and Machine Intelligence from 1987 to 1989, Director of the NATO Advanced Research Workshop on Multisensor Fusion for Computer Vision, Grenoble, France, 1989; Chairman of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1993, and President of the International Association for Pattern Recognition from 1992 to 1994. He currently serves as IEEE Computer Society representative to the IAPR.