To appear in International Journal of Computer Vision, 1998.

New Visual Invariants for Terrain Navigation Without 3-D Reconstruction

Gin-Shu Young[1,2], Martin Herman[1], Tsai-Hong Hong[1], David Jiang[1,2], and Jackson C. S. Yang[2]

[1]National Institute of Standards and Technology (NIST), Bldg 220, Rm B124, Gaithersburg, MD 20899

[2]Robotics Laboratory, Department of Mechanical Engineering, University of Maryland, College Park, MD 20742

Abstract

For autonomous vehicles to achieve terrain navigation, obstacles must be discriminated from terrain before any path planning and obstacle avoidance activity is undertaken. In this paper, a novel approach to obstacle detection has been developed. The method finds obstacles in the 2-D image space, as opposed to 3-D reconstructed space, using optical flow. Our method assumes that both non-obstacle terrain regions, as well as regions with obstacles, will be visible in the imagery. Therefore, our goal is to discriminate between terrain regions with obstacles and terrain regions without obstacles. Our method uses new visual linear invariants based on optical flow. Employing the linear invariance property, obstacles can be directly detected by using reference flow lines obtained from measured optical flow. The main features of this approach are: (1) 2-D visual information (i.e., optical flow) is directly used to detect obstacles; no range, 3-D motion, or 3-D scene geometry is recovered; (2) knowledge about the camera-to-ground coordinate transformation is not required; (3) knowledge about vehicle (or camera) motion is not required; (4) the method is valid for the vehicle (or camera) undergoing general six-degree-of-freedom motion; (5) the error sources involved are reduced to a minimum, because the only information required is one component of optical flow. Numerous experiments using both synthetic and real image data are presented. Our methods are demonstrated in both ground and air vehicle scenarios.

1 Introduction

For autonomous vehicles to achieve terrain navigation, obstacles must be discriminated from terrain before any path planning and obstacle avoidance activity is undertaken. Obstacles are defined as any regions in space where a vehicle should not or cannot traverse, such as protrusions (objects lying on top of the terrain), depressions (potholes, ditches, gullies in the terrain), or steep terrain

(Figure 1). This paper describes a simple, fast, and general method for obstacle detection for ground vehicles or air vehicle landings. The vehicle may move under general motion, i.e., arbitrary translation and rotation.

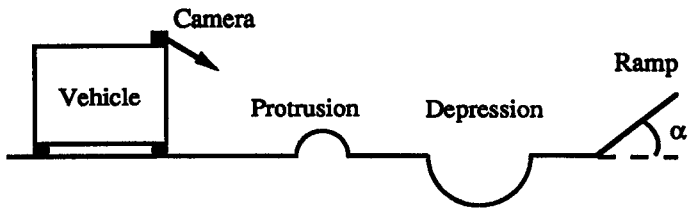


Figure 1: Terrain with obstacles.

For many applications in computer vision, it is important to recover range, 3-D motion, and/or 3-D scene geometry from a sequence of images [17] [43] . However, many vision-based behaviors can be achieved without recovery of such information [3] [15] [19] [20] [36] [37] . These behaviors can be achieved by extracting relevant 2-D information from the imagery and using this information directly without 3-D reconstruction.

The advantages of this 2-D based approach include greater simplicity and speed. The approach is simpler because fewer hypotheses, sensor measurements, and calibrations need to be performed when using 2-D information rather than 3-D information. The reason is that the problem of converting 2-D image information to 3-D world information requires either hypotheses about the world to be made (e.g., smoothness hypotheses, lighting hypotheses, object hypotheses), calibrations to be made (e.g., camera-to-ground transformations, stereo calibrations, inertial navigation system calibrations), or many sensor measurements to be made (e.g., inertial navigation system measurements). The 2-D based approach is faster because it generally requires fewer computations, such as coordinate transformations or object model fitting.

To perform obstacle detection, either active sensors (such as laser scanners, radar, and ultrasonics) or passive sensors (cameras) may be used. The use of passive, instead of active, sensors can eliminate radiation, reduce cost, and increase flexibility for many applications. Optical flow, used by many biological creatures for navigation [2] [32] , can provide very powerful information for vision-based navigation, during both teleoperated low data rate driving and autonomous driving [18] .

In this work, new visual invariants are developed as a tool to detect obstacles directly from optical flow. Our method assumes that both non-obstacle terrain regions, as well as regions with obstacles, will be visible in the imagery. Therefore, our goal is to discriminate between terrain regions with obstacles and terrain regions without obstacles.

The visual invariants we develop involve the mapping of points that lie on any straight-line segment in 3-D space into an image-based space, i.e., a space whose coordinate axes represent parameter values extracted from the image domain. There are certain image-based spaces such that straight lines in these spaces are mapped *only* from straight lines in 3-D space.[1] Such a mapping is described as invariant for linear relationships, or simply linearly invariant, because linear relationships are always preserved. For example, we show below that a straight-line segment $\overline{P_1P_2}$ in 3-D space (Figure 2(a)) always maps to a straight line segment $\overline{F_1F_2}$ in the image-based space whose coordinates are x, $\dot{y}$ (Figure 2(b)) where x is the image position along the image line y and $\dot{y}$ (Figure 2(a)) is the y component of optical flow. If a point P_5 in 3-D space (Figure 2(a)) lies on the extended line segment $\overline{P_1P_2}$, then the image point F_5 (Figure 2(b)) corresponding to P_5 *must* lie on the extended line segment $\overline{F_1F_2}$. In Sections 4 and 5, we demonstrate that this type of visual invariant allows us to detect obstacles using optical flow.

Figure 3(a) shows a portion of terrain with a protrusion and depression which is visible in the camera. The visual information for five points (P_1, P_2, P_3, P_4, and P_5) lying on the terrain is represented in the image-based space $\dot{y}$ vs. x (Figure 3(b)). The reference flow line (Figure 3(b)) obtained from F_1 and F_2 corresponds to a reference space line in 3-D space (Figure 3(a)). The deviation (or difference) between the measured value $\dot{y}$ (F_1, F_2, F_3, F_4, and F_5) and the $\dot{y}$ value of the reference flow line for each image position x is calculated. Figure 3(c) is a plot of the deviation. A positive deviation corresponds to a protrusion while a negative deviation corresponds to a depression. Notice in Figure 3(a) that the reference line in 3-D space, formed by points P_1, P_2 and P_5, need not lie on a linear scene feature, but instead may lie in a flat region in the scene. Also, the portion of the line between P_2 and P_5 does not lie in any scene surface. It is shown in Section 4 that if the

1. As will be described below, the straight line in 3-D space need not correspond to a straight-line feature in the scene. It may be an imaginary 3-D line, a portion of which lies on a relatively flat surface region.

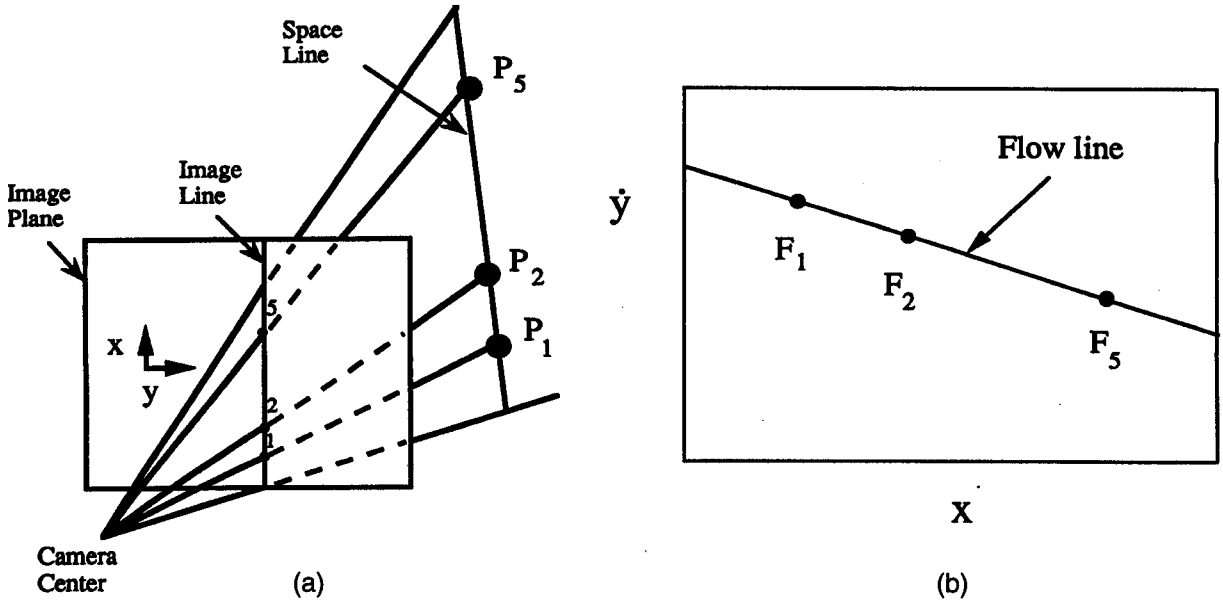


Figure 2: (a) Space line and image line. (b) Flow line in the image-based space $\dot{y}$ vs x.

linear relationship is not maintained in the image-based space, then it is difficult to detect obstacles in that space. Therefore, the mapping must be linearly invariant to be useful. The features of this method are:

1. 2-D visual information (i.e., optical flow) is directly used to detect obstacles: no range, 3-D motion, or 3-D scene geometry is recovered;

2. no information about the pose of the camera relative to the ground is required, reducing the amount of camera calibration required;

3. no information about the vehicle (or camera) motion is required, eliminating the need for extra sensors and calibration of the camera pose relative to these sensors;

4. arbitrary camera motion (both translation and rotation) is allowed, making the method completely general.

In Section 2, a background and discussion of previous work is presented. Visual linear invariants are developed in Section 3. Section 4 shows how to detect protrusions and depressions directly from optical flow without 3-D reconstruction. A detailed algorithm for obstacle detection is given. A number of experiments using both synthetic and real indoor and outdoor scenes are presented in

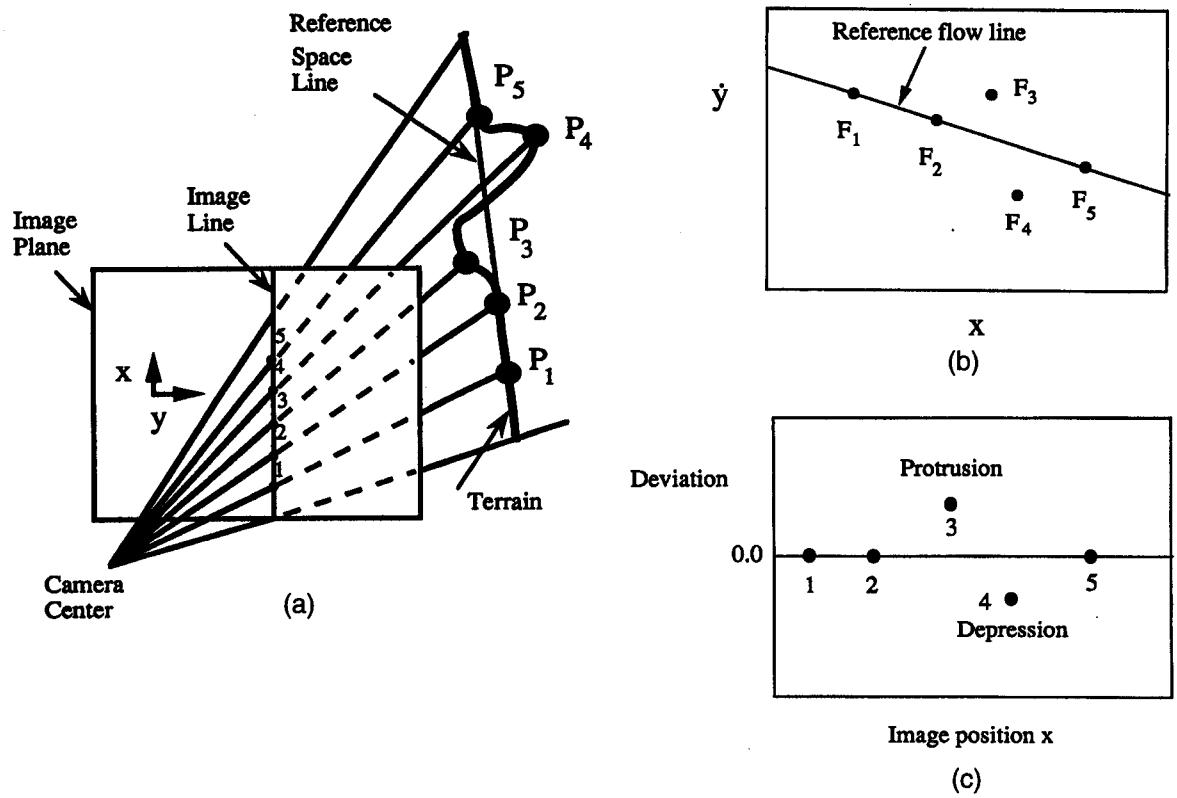


Figure 3: (a) Visible terrain along an image line. (b) Reference flow line in the image-based space $\dot{y}$ vs. x. (c) Obstacle detection without 3-D reconstruction.

Section 5.

The method described here deals with obstacle detection, not obstacle avoidance. The latter requires a determination of spatial position, and perhaps size, of the obstacle. We propose in the future to use a flow or image divergence approach to determine the temporal distance of the detected obstacle. Such an approach does not perform 3-D reconstruction. This issue, along with conclusions and other future work, is discussed in Section 6.

2 Previous Work

A number of obstacle detection methods have been developed in the past (e.g., [5] [9] [10] [11] [12] [13] [14] [16] [21] [31] [34] [38] [39] [40] [41] [44]). Range information is often employed to solve this problem. This information may be obtained from active sensors, stereo cameras, optical flow, etc. *A priori* knowledge is required by most existing methods. Such knowledge may include sensor-to-ground coordinate transformations, sensor motion, model optical flow

fields, road models (or maps), etc. Errors in *a priori* knowledge result in errors in the output. Also, the requirement for *a priori* knowledge makes these systems less flexible.

Several obstacle detection methods based on optical flow have been developed. Sridhar et al. [40] investigated obstacle detection for rotorcraft low altitude flight. The obstacle detection problem is posed as the problem of finding range to all objects in the field of view, and range is obtained from optical flow. Bhanu et al. [5] presented an inertial sensor integrated optical flow technique for motion analysis in which range is extracted from optical flow for obstacle detection. The method by Hoff and Sklair [21] detects landing hazards for a descending spacecraft. They develop an algorithm using range information retrieved from optical flow with known camera motion.

Without 3-D reconstruction, time-to-collision estimated from flow divergence, changes of image gradients over time, etc. have been used for obstacle avoidance [3] [9] [10] [11] [26] [33] [37]. These methods, however, do not consider the problem of finding obstacles when both obstacles and non-obstacle terrain are visible in the image. The portion of the terrain nearest the observer will have the smallest time-to-collision values, but may not necessarily be an obstacle region to be avoided.

Some researchers have considered this problem of discriminating obstacles from terrain. Enkelmann [14] detects obstacles by evaluating the difference between calculated optical flow and estimated model flow. The estimated model requires knowledge of the focus of expansion (FOE), the transformation matrix between the camera and vehicle coordinate systems, and the camera motion. In addition, this method works only for a camera that undergoes pure translation parallel to a planar surface. Tistarelli and Sandini [42] detect obstacles by evaluating the difference between calculated optical flow and a reference flow map. The method assumes pure translational camera motion parallel to the planar surface that gives rise to the reference flow map. It requires knowledge of the FOE and the camera motion. Raviv [35] detects obstacles using an optical-flow-based invariant with the assumption of an observer who undergoes pure translational motion parallel to a planar surface. Mallot et al. [30] detect discrete obstacles by the use of inverse perspective mappings. This method requires a coordinate transform between the camera and the ground plane and, again, assumes that the camera moves in the horizontal plane under pure translation. Sandini et al. [37] have considered the problem of discriminating obstacles from ground without 3-D reconstruction

using binocular disparity maps. The similarity between our method and theirs is the use of reference maps to represent the ground. Of course, our method uses flow rather than binocular disparity.

The work described above may be characterized by the following:

1. Range information extracted from optical flow, stereo, or active range sensors is often employed to detect obstacles.

2. Many approaches in which obstacles are detected directly from optical flow do not consider the problem of discriminating obstacles from terrain.

3. For approaches that directly use optical flow to discriminate between obstacles and terrain, either the observer's motion is restricted to pure translation or *a priori* knowledge, such as coordinate transformations, camera motion, or model optical flow fields, are required.

Our method is unique because it uses optical flow to discriminate between obstacles and terrain without 3-D reconstruction, for arbitrary camera motion, and without knowledge of camera-to-ground coordinate transformations or camera motion. Our method is, therefore, a general method which requires relatively little calibration or *a priori* knowledge. This means that the error sources involved are reduced to a minimum because the only information required is optical flow.

To demonstrate the advantage of reducing the error sources, consider a very simple example of converting optical flow to range using the measured motion of the camera. Figure 4(a) shows the 2-D scenario of a spherical camera undergoing pure translation along its z axis toward a flat wall. The optical flow $\dot{\theta}$ at any point θ in the image is [1] :

$$\dot{\theta} = \frac{|\dot{\mathbf{r}}| \sin\theta}{r} \tag{1}$$

where r is the true range to the wall and $\dot{\mathbf{r}}$ is the camera velocity.

To recover range from optical flow, the speed and heading of the camera is determined. Consider the errors due only to an angular error ϕ in the heading estimate (Figure 4(b)). (Assume that the speed estimate is not in error.) Then, the recovered range estimate $\hat{r}$ is

$$\hat{r} = \frac{\sin(\theta - \phi)}{\sin\theta} r \tag{2}$$

and the recovered z component, $\hat{d}$, of the range estimate is

$$\hat{d} = \frac{\sin(\theta - \phi)}{\sin\theta} d \tag{3}$$

where d is the true z component of range (also referred to as depth). Figure 4(c) shows the percent error in the depth estimate as a function of small errors ϕ in the camera heading. Notice that errors can be significant for values of θ less than $10°$, and even for values of θ up to $30°$ if heading errors are about $1°$. If we can detect obstacles directly from flow, without explicitly recovering range or depth, then one source of error - heading estimate error - is eliminated, leading to potentially more accurate obstacle detection.

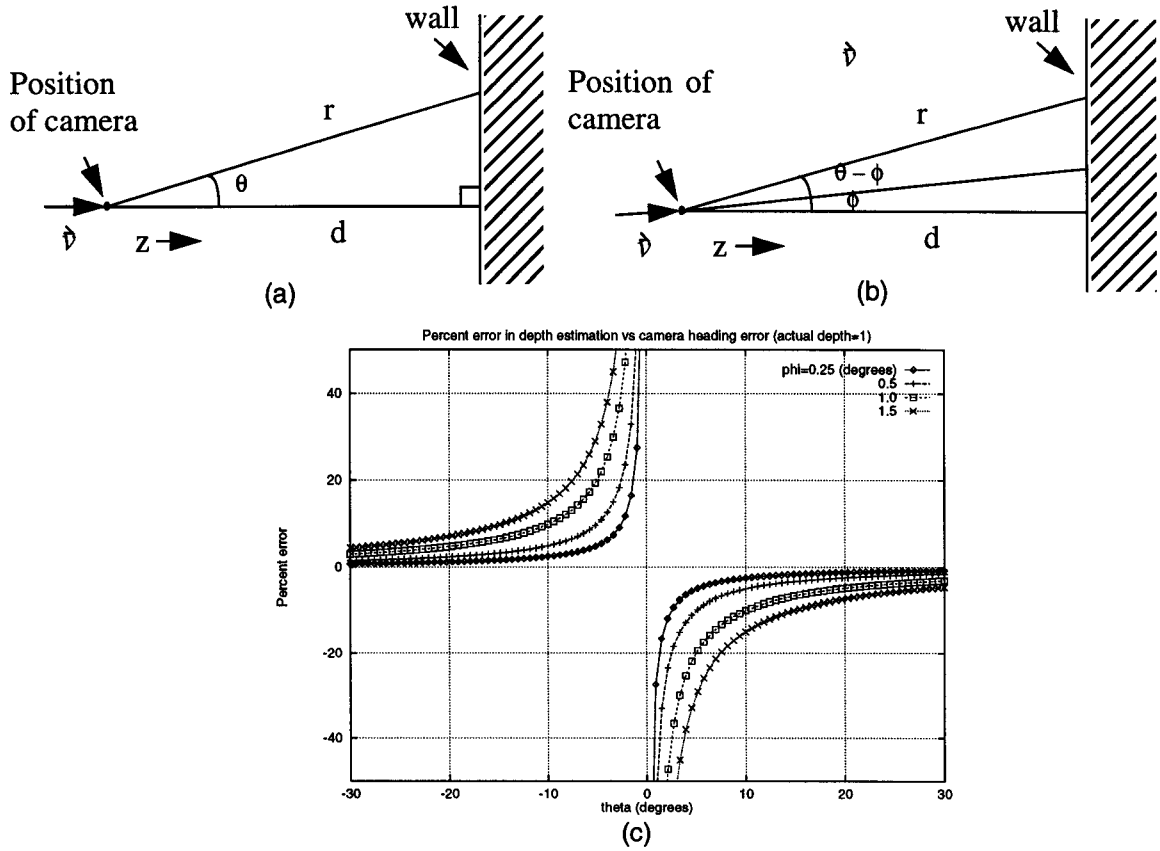


Figure 4: (a) A spherical camera translating toward a wall. (b) An error ϕ in the heading estimation results in errors in the depth estimation. (c) Percent errors in depth estimation. Negative (positive) percentages indicate that the depth estimate is smaller (larger) than true depth.

3 Visual Invariants

3.1 Background

Coordinate frames

Two coordinate frames that are important in our approach are the camera coordinate frame and the coordinate frame attached to a line in space. Consider first the arbitrary line segment $\overline{AB}$ in space and its projection onto the image plane, line segment $\overline{ab}$ (Figure 5). Define the image x axis to be parallel to line segment $\overline{ab}$. A coordinate frame c attached to the camera is chosen as follows:

1. Let the origin o_c be at the camera focal point.

2. Let the z_c axis be the optical axis.

3. Choose x_c to be parallel to the image x axis.

4. Choose y_c to obey the right hand rule.

5. Choose the image y axis to be parallel to y_c.

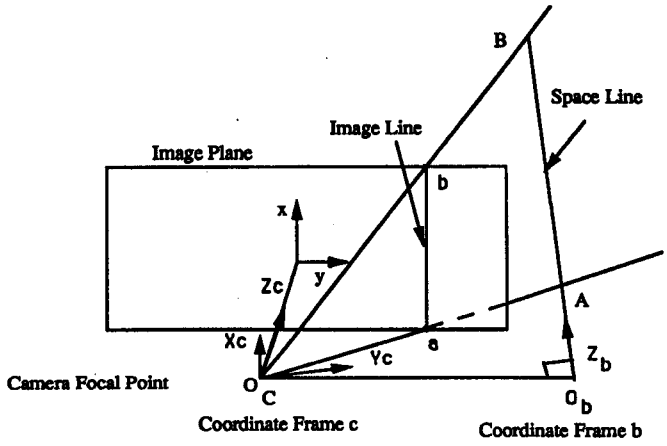


Figure 5: Definition of two coordinate frames.

A coordinate frame b is then affixed to the line $\overline{AB}$ as follows:

1. Let the origin O_b be the point lying on the extended line $\overline{AB}$ with the shortest distance from the camera focal point O_c.

2. Let the Z_b axis be line $\overline{AB}$.

3. Choose X_b and Y_b arbitrarily as long as the right hand rule is obeyed.

Transformation matrix

A point P in the scene can be transformed from frame b to frame c by the equation:

$$\begin{Bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{Bmatrix} = H_b^c * \begin{Bmatrix} X_b \\ Y_b \\ Z_b \\ 1 \end{Bmatrix} \tag{4}$$

where (X_c, Y_c, Z_c) and (X_b, Y_b, Z_b) are the coordinates of point P in frames c and b, respectively, and

$$H_b^c = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

represents a 4x4 transform matrix from frame b to frame c. Note that for a moving camera, at each instance of time, H_b^c is constant for all points in the scene.

Basic equations

The visual invariants introduced here are based on optical flow $\bar{F}$, which can be expressed as:

$$\bar{F}(x, y, t) = (\dot{x}(x, y, t), \dot{y}(x, y, t)) = F(x, y, t)\bar{u}_F \tag{6}$$

where (x, y) is the image position, t is the instance of time, $\dot{x}$ and $\dot{y}$ are the components of optical flow, and F and $\bar{u}_F$ are the magnitude and unit directional vector of optical flow, respectively.

From the pinhole camera model, if we let the focal length be unity, the image position (x, y) is

$$x = \frac{X_c}{Z_c} \tag{7}$$

$$y = \frac{Y_c}{Z_c} \tag{8}$$

As defined earlier, line $\overline{AB}$ coincides with the Z_b axis (Figure 5). Therefore any points lying on line $\overline{AB}$ always have

$$X_b = Y_b = 0 \tag{9}$$

For a point P lying on line $\overline{AB}$, we use Equations (4), (5) and (9) to express the relationship between its coordinates in frames c and b as

$$X_c = h_{13}Z_b + h_{14} \tag{10}$$

$$Y_c = h_{23}Z_b + h_{24} \tag{11}$$

$$Z_c = h_{33}Z_b + h_{34} \tag{12}$$

With Equations (7), (10) and (12), $1/Z_c$ can be expressed in terms of x as the following:

$$\frac{1}{Z_c} = \frac{(h_{13} - xh_{33})}{(h_{34}h_{13} - h_{14}h_{33})} \tag{13}$$

Note that $(h_{34}h_{13} - h_{14}h_{33})$ cannot equal zero because Z_c cannot equal zero for any 3-D point visible in the camera.

3.2 Derivation

The equations for optical flow due to general camera motion (arbitrary translation and rotation) in a stationary environment are [28] :

$$\dot{x} = \frac{1}{Z_c}(-T_X + xT_Z) + (xy\omega_X - (1 + x^2)\omega_Y + y\omega_Z) \tag{14}$$

$$\dot{y} = \frac{1}{Z_c}(-T_Y + yT_Z) + ((1 + y^2)\omega_X - xy\omega_Y - x\omega_Z) \tag{15}$$

$$F = (\dot{x}^2 + \dot{y}^2)^{\frac{1}{2}} \tag{16}$$

where Z_c is the depth of the object in the environment relative to the camera, (T_X, T_Y, T_Z) and $(\omega_X, \omega_Y, \omega_Z)$ are the translational and rotational motion of the environment coordinate system relative to the camera, and F is the magnitude of optical flow. Note that, for each instance of time, (T_X, T_Y, T_Z) and $(\omega_X, \omega_Y, \omega_Z)$ are constants for all points in the stationary environment.

Using Equation (13), the following linear relationship can be obtained from Equation (15) for all image points lying on line $\overline{ab}$ (i.e., y=constant) that arise from points in the scene lying on line $\overline{AB}$ (Figure 5):

$$\dot{y} = a_1 + a_2 x \tag{17}$$

where

$$\begin{aligned} a_1 &= h_{13}a_3 + (1 + y^2)\omega_X \\ a_2 &= -h_{33}a_3 - y\omega_Y - \omega_Z \\ a_3 &= \frac{(-T_Y + yT_Z)}{(h_{34}h_{13} - h_{14}h_{33})} \end{aligned} \tag{18}$$

As explained earlier, the value $(h_{34}h_{13} - h_{14}h_{33})$ cannot equal zero. For each instance of time, h_{ij} (the components of H_b^c) and (T_X, T_Y, T_Z) and $(\omega_X, \omega_Y, \omega_Z)$ are constants. Since y = constant, for each instance of time, the values a_1, a_2, and a_3 are also constants for all points on line $\overline{AB}$. Equation (17) represents a line in the $\dot{y}$ vs. x image-based space corresponding to a line in 3D space. This is a visual linear invariant. Note that, in principle, the line in the $\dot{y}$ vs. x image-based space can be estimated from two points (say, $(x_1, \dot{y}_1)$ and $(x_2, \dot{y}_2)$). This means that specific knowledge about the transformation matrix and camera motion is not required.

If we were to combine Equations (16), (19), and (17), then the resulting F as a function of x would be nonlinear. This is *not* a visual invariant for linear relationships.

If we were to combine Equations (13) and (14), the following nonlinear relationship can be ob-

tained for all image points lying on line $\overline{ab}$ (i.e., y=constant) that arise from points in the scene lying on line $\overline{AB}$ (Figure 5):

$$\dot{x} = b_1 + b_2 x + b_3 x^2 \tag{19}$$

where b_1, b_2, b_3 are constants. Equation (19) represents a quadratic curve in the $\dot{x}$ vs. x image-based space corresponding to a line in 3-D space. This is *not* a visual invariant for linear relationships. However, consider the special case of camera motion along the x axis, which corresponds to sideward-looking camera motion.[1] For this case, $T_Y = T_Z = \omega_X = \omega_Y = \omega_Z = 0$. The component $\dot{y}$ of optical flow is zero, but now

$$\dot{x} = a_1 + a_2 x \tag{20}$$

where

$$\begin{aligned} a_1 &= h_{13} a_3 \\ a_2 &= -h_{33} a_3 \\ a_3 &= \frac{(-T_X)}{(h_{34} h_{13} - h_{14} h_{33})} \end{aligned} \tag{21}$$

This is a visual invariant for linear relationships.

3.3 Camera motion in horizontal plane

As described above, a straight line in 3-D space can be mapped into either a straight line or a curve depending on the camera motion and the image-based space used for the mapping. However, a mapping is always linearly invariant for certain image-based spaces ($\dot{y}$ vs. x, $\dot{x}$ vs. y). Further, from Equations (17) and (18), a straight line in 3-D space can always be mapped into a straight line with slope = 0 if $a_2 = 0$ (or $h_{33} = \omega_Y = \omega_Z = 0$). This will result in simple computations.

Consider the case with the following two conditions:

1. The terrain is flat and the image line x (or x_c) is parallel to the terrain. The space line Z_b

1.This is really an arbitrary sideward-looking camera motion in the sense that the camera x axis changes according to the particular image line of interest (Figure 5).

whose projection is the image line x lies on the terrain and is perpendicular to the optical axis Z_c (i.e., $h_{33} = \cos\theta_{Z_cZ_b} = 0$ since $\cos\theta_{X_cZ_b} = 1$, where $\theta_{Z_cZ_b}$ is the angle between Z_c and Z_b and $\theta_{X_cZ_b}$ is the angle between X_c and Z_b (see Figure 5)).

2. Any translational motion and only rotational motion about X_c are allowed (i.e., $\omega_Y = \omega_Z = 0$).

This corresponds to a camera moving over a flat terrain and the image line of interest is horizontal (e.g., a scan line). Only rotation about the horizontal axis (e.g., camera tilt) is permitted. In this case, we see from Equations (17) and (18) that for each instance of time, all image points lying on the horizontal image line have constant values

$$\dot{y} = a \tag{22}$$

where

$$a = \frac{(-T_Y + yT_Z)}{(h_{34})} + (1 + y^2)\omega_X \tag{23}$$

Different horizontal image lines will have different constant values for a.

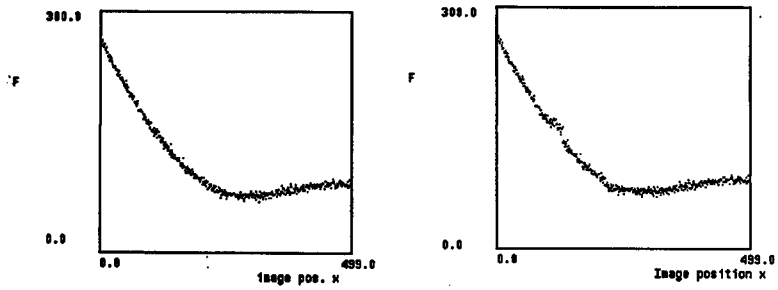
4 OBSTACLE DETECTION

In this section, we show how to detect protrusions and depressions using a purposive and direct approach (i.e., directly from optical flow without 3-D reconstruction). The method employs the properties of visual linear invariants [46] .

4.1 Visual linear invariants for obstacle detection

As described in Section 3, mappings are linearly invariant only for certain image-based spaces. We now show that these invariant mappings are very useful for obstacle detection. Consider the mapping to the F vs. x image-based space, which is full-flow magnitude F as a function of position x on the image line under consideration. This mapping is not linearly invariant. Figures 6(a), (b) show the results of a simulation for two different types of terrain. The simulation involves 5% noise added to synthetically generated optical flow under general camera motion. Although no obvious

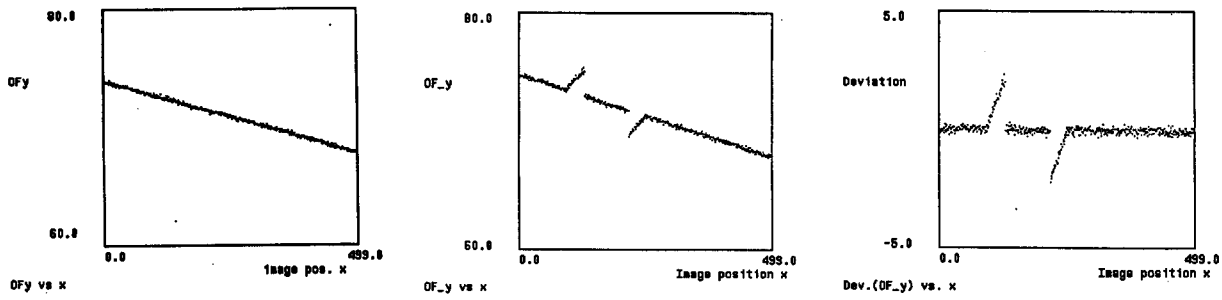
differences can be observed between Figures 6(a) and (b), in Figure 6(a) the terrain is flat and without obstacles, while in 6(b) the terrain has two obstacles, one protrusion and one depression. We see that detecting obstacles can be very difficult in an image-based space that does not have the linear invariance property.



(a) (b)

Figure 6: (a) F vs. x without obstacles. (b) F vs. x with two obstacles.

On the other hand, when an image-based space with the property of linear invariance is used, obstacle detection becomes easy and straightforward. Consider the mapping to the $\dot{y}$ vs. x image-based space, which is linearly invariant (Equation (17)) under general camera motion. Figures 7(a) and (b) show results for the same simulation, using the same camera motion and terrain, as in Figure 6. Figure 7(a) appears to coincide with a reference flow line, implying that it is obtained from a terrain which is flat (at least in one dimension). Points that do not lie on the reference line result from protrusions or depressions in the terrain. Thus, in Figure 7(b), there are two obstacles, one a protrusion and the other a depression. Figure 7(b) can further be mapped into Figure 7(c) where the reference line is horizontal. Figure 7(c) plots the difference between the actually measured value $\dot{y}$ and the $\dot{y}$ value of the reference line for each x value. A protrusion or depression can easily be detected in 7(c). Detailed experimental results are shown later.



(a) (b) (c)

Figure 7: (a) $\dot{y}$ vs. x without obstacles. (b) $\dot{y}$ vs. x with two obstacles. (c) $\Delta\dot{y}$ vs. x with two obstacles.

4.2 Algorithm

To apply the property of linear invariance to obstacle detection, four steps are involved.

Step 1: Selection of a straight line in the image.

The method will work for any arbitrary line in the image, no matter what its position or orientation. However, the line should intersect an image feature of interest, e.g., a potential obstacle. The chosen image line need not correspond to a linear feature in the scene. In principle, many image lines can be processed in parallel.

Step 2: Estimation of the reference flow line for the selected image line.

The reference flow line in the image-based space, say $\dot{y}$ vs. x for an image line y, corresponds to a reference space line in 3-D space. The latter reference line and the camera focal point define a plane in space (Figure 3(a)). The intersection of this plane with objects and terrain in the environment defines a set of curves lying in the plane which are visible in the camera. Deviations between these curves and the 3-D reference line are represented in the image-based space (Figure 3(c)). The reference flow line can be arbitrarily chosen from the regions represented by these curves, but it should probably arise from a surface in the environment such that deviations from this surface represent protrusions and depressions. On a road, for example, the 3-D reference line should probably lie on the road surface. For a vertical image line, the reference flow line might be obtained from the measured optical flow on the image line located at the bottom of the image, which would probably correspond to points on the ground surface near the vehicle. For an arbitrary image line, the reference flow line can be obtained by fitting a straight line to the measured optical flow selected at some image positions which correspond to the reference region in the scene; the 3-D reference line will then lie in the surface that forms the reference region. Note that only the component of optical flow normal to the image line is used. In principle, only two points are required to estimate the reference flow line.

Step 3: Computation of the deviation.

The deviation (or difference) between the reference line obtained in **step 2** and the measured flow

at all image positions lying on the image line chosen in **step 1** is computed.

Step 4: Representation of obstacle regions.

The computed deviation in **step 3** is used to detect obstacles. Points on the image line with deviations larger than some threshold value represent obstacles.

To detect obstacles easily, an image-based space that has the visual linear invariance property should be used. For arbitrary camera motion, the proper image-based space is $\dot{y}$ vs. x (or $\dot{x}$ vs. y) for any image line y=constant (or x=constant) if $\dot{y}$ (or $\dot{x}$) is not equal to zero. For the case where $\dot{y}$ (or $\dot{x}$) is zero for all points on an image line, $\dot{x}$ vs. x (or $\dot{y}$ vs. y) should be used.

With this method, only one component of optical flow is needed. Information such as specific knowledge of vehicle (or camera) motion, or knowledge of the coordinate transformation between the camera and the ground, is not required. Therefore, the method reduces error sources to a minimum because it employs minimum information. The approach is simple because obstacles are detected directly in the image-based space, without performing 3-D reconstruction. There is no assumption of a terrain model; this method can, therefore, be used to navigate indoors or outdoors on roadways or natural terrain. This method can also be used for air vehicles undergoing general six-degree-of-freedom motion while landing on either known or unknown terrain.

4.3 Control Strategies

The discussion thus far has been concerned only with detecting obstacles in single image lines. There may be several ways in which this basic idea can be applied. One way involves choosing a set of image lines that covers the whole image. For example, all rows in the image may be processed in parallel. (In the experiment in Section 5.6, all rows in the middle portion of the image are processed.) In this way, all obstacles will be found. However, care would need to be taken so that rows that intersect the sky are not processed, since the method has been developed for discriminating obstacles from terrain. Another approach involves processing all columns in parallel. This would be useful if the bottom portion of the image can be assumed to correspond to the ground surface near the vehicle, since this ground surface should be used to derive the reference flow lines.

In another approach, a single horizontal strip in the image is processed. This strip may be formed from a single image row, or from a small number of multiple contiguous rows (see Section 5.5 and Figures 21(a), 22(a)). As the vehicle moves forward, this strip is "swept" over the scene ahead. All obstacles in front of the vehicle at a certain distance will be detected. The higher up in the image the strip lies, the further away are the obstacles that will be detected.

5 Experimental Results

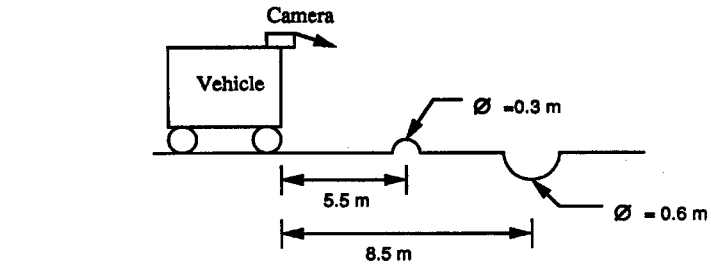
In this section, we present the results of several experiments demonstrating the simplicity and usefulness of visual linear invariants applied to obstacle detection. These experiments include ground and air vehicles. Both synthetic and real indoor and outdoor scenes are considered. To extract optical flow from real imagery, we have used several different algorithms. Only the components of flow normal to the selected image lines are used.

5.1 Ground and Air Vehicle Simulations

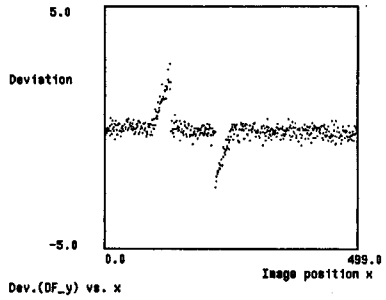
The first experiment simulates a ground vehicle moving over terrain with a bump and a pothole (Figure 8(a)). Synthetic data are used with three different levels of noise: 5%, 10%, and 15%. The noise is generated randomly using a zero-mean Gaussian distribution. The generated noise is added to the magnitude of the perpendicular component of flow obtained through simulation. This noise represents the uncertainty value of the flow. Only one image line (a vertical line) is used in this experiment. The bump is a semicircle 5.5 m ahead of the camera, with a height of 0.3 m above the flat terrain. The pothole is a semicircle 8.5 m ahead of the camera, with a depth of 0.6 m below the terrain. The camera is mounted on top of the vehicle (2 m above the ground) and moves under general motion with (T_X = -927, T_Y = -4, T_Z = 2853 mm/s) and (ω_X = 0.05, ω_Y = 0.05, ω_Z = 0.05 rad/s).

The results with 5% noise added to the synthetic flow have already been shown in Figure 7(c), where two obstacles, a protrusion and a depression, can easily be detected. The results with noise of 10% and 15% are shown in Figures 8(b), (c). Again, the obstacles are easy to detect.

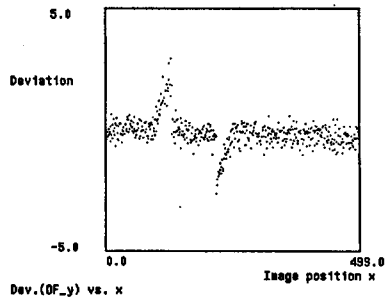
The next experiment simulates an air vehicle moving over terrain with multiple rectangular protrusions (Figure 9(a)). The first protrusion is 45 m ahead of the camera at a height of 1 m above the flat terrain. The second protrusion is 63 m ahead of the camera at a height of 1 m.The camera is



(a)



(b)



(c)

Figure 8: (a) Side view of terrain with bump and pothole. (b) $\Delta\dot{y}$ vs. x (10% noise). (c) $\Delta\dot{y}$ vs. x (15% noise).

mounted on the bottom of the air vehicle (20 m above the ground) and moves under general motion with (T_X = −13750, T_Y = −40, T_Z = 48050 mm/sec) and (ω_X = 0.1, ω_Y = 0.5, ω_Z = 0.05 rad/sec).

As before, three different levels of noise, 5%, 10%, and 15% are added to the synthetic optical flow. The results, presented in Figures 9(b), (c), (d), show that the obstacles are easy to detect. (With 15% noise, we begin to see some difficulty in the detection.)

5.2 Table-Top Setup

This experiment involves detecting a depression in a flat surface using real images. Figure 10(a) shows the table-top setup, in which the camera is mounted on a linear positioning table and moves along a direction perpendicular to the camera's optical axis. The motion of the camera is along the x axis with speed T_X = 22.5 mm/s. A flat surface with a square depression of 50.8 mm is at a depth of 406.4 mm from the camera. The depression is located in the middle of the image shown in Figure 10(b). Since $\dot{y} = 0$, we use the $\dot{x}$ vs. x image-based space, which has the property of linear invariance.

Only the scan line labeled in Figure 10(b) is used. Optical flow at each pixel is obtained using a

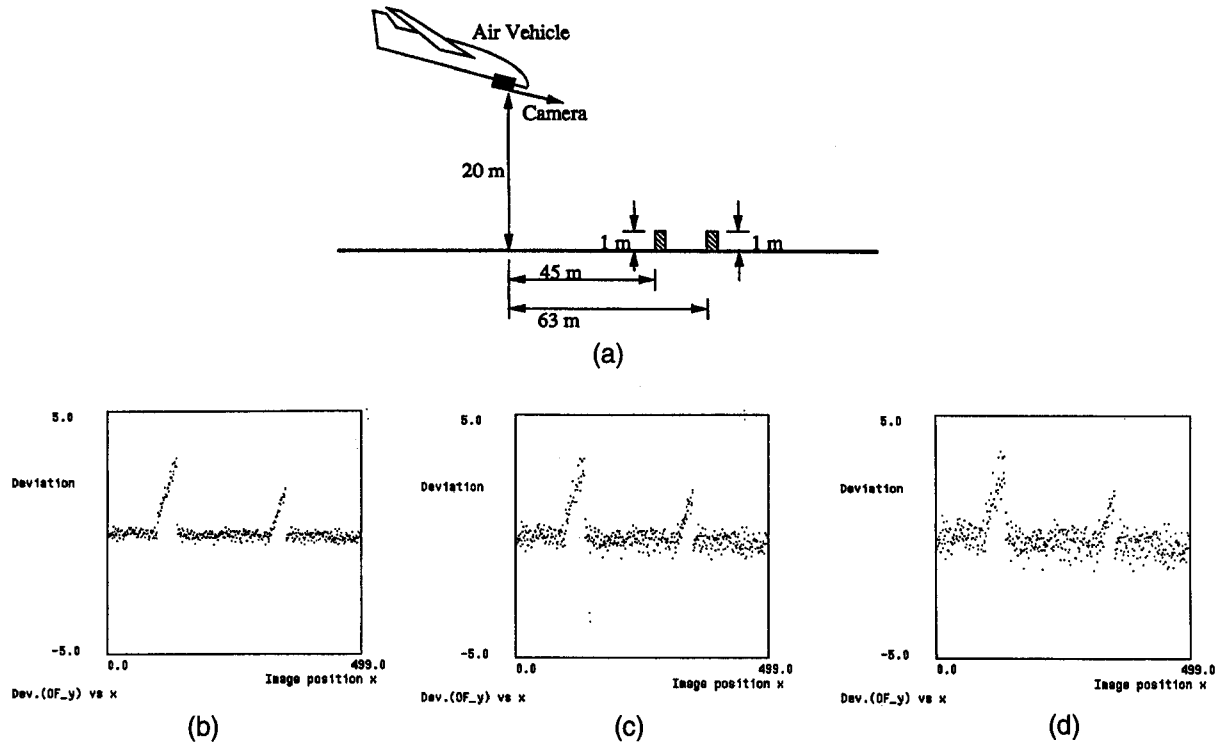


Figure 9: (a) Side view of terrain with two protrusions. (b) $\Delta\dot{y}$ vs. x (5% noise). (c) $\Delta\dot{y}$ vs. x (10% noise). (d) $\Delta\dot{y}$ vs. x (15% noise).

correlation method [22] . The values of optical flow $\dot{x}$ at image positions 40 through 199 are shown in Figure 10(c). The reference flow line is obtained by fitting a straight line to the data points near image position 40. The deviation of $\dot{x}$ from the reference flow line is shown in Figure 10(d). A depression can easily be detected in Figures 10(c), (d).

5.3 Yosemite Flyby Sequence

In this experiment, we use realistic synthetic images of a 3-D natural outdoor scene (Figure 11), thus allowing us to compare the results with ground truth. Our goal is to detect the mountains, which are hazardous regions for an air vehicle, in the Yosemite flyby sequence (obtained from Barron et al. [4] , courtesy of Lynn Quam). This sequence was created from aerial photos and terrain maps to simulate an aerial flyby. The sky and clouds are artificial.

Two regions have been selected for obstacle detection: a horizontal strip consisting of rows 200 to 220 and a vertical strip consisting of columns 270 to 290. Although a single horizontal or vertical line may be used, a range of rows or columns produces better results because the effects of noise are reduced after median filtering.[1] However, the speed of the system decreases as the number of

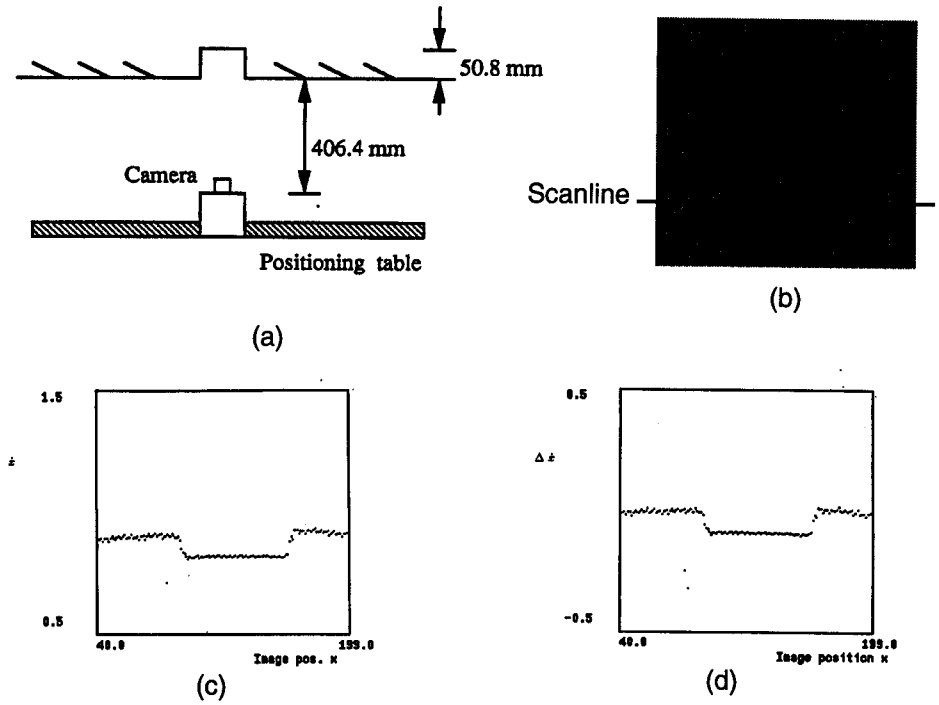


Figure 10: (a) Top view of table-top setup. (b) Original image. Only data from the single scanline indicated are used. (c) $\dot{x}$ vs. x. (d) $\Delta\dot{x}$ vs. x.

lines increases. Thus, care must be taken in selecting the number of lines.

The results of two optical flow algorithms used in this experiment are compared. These algorithms are Liu, et al.'s [27] gradient-based algorithm (henceforth referred to as Liu's algorithm) and Camus's [7] [8] correlation-based algorithm. Liu's algorithm produces very accurate optical flow but requires a significant amount of computation time, whereas Camus's algorithm produces reasonably accurate flow with minimal computation time. Both algorithms could be used to complement each other in a real-time obstacle avoidance system. Camus's algorithm could be used to detect close obstacles, while Liu's algorithm could be used for more distant obstacles.

The true flow field is shown in Figures 12(a), (b). Liu's algorithm produces a very accurate flow output (Figure 12(c), (d)) while Camus's algorithm produces a reasonable flow output (Figures 12(e), (f)) at approximately 1/60-th the time of Liu's algorithm. In both cases, flow is also produced

1. We apply two median filters. The first is a reference line median filter that filters the data points used to generate the reference line. The second is a deviation line median filter to smooth the deviation output. Increasing the respective window sizes helps to smooth the data but increases the computation time. In most cases, the default setting of a reference line median filter window size of 3x3 and a deviation median filter window size of 3x3 produces good results.

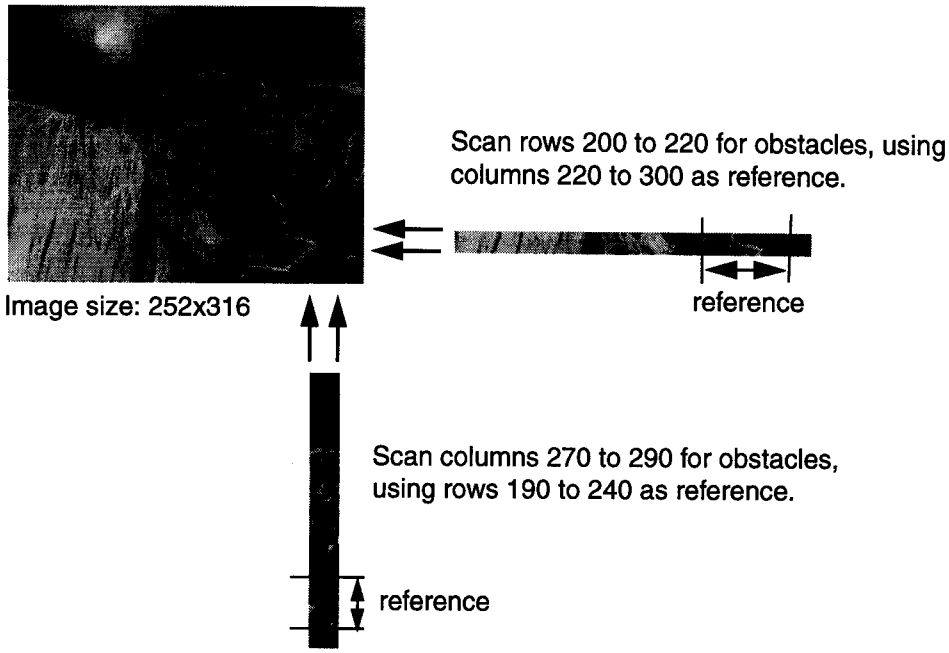


Figure 11: Yosemite flyby sequence.

for the synthetic clouds, something that is not depicted in the true flow. This flow can produce difficulties in obstacle detection as we shall subsequently see.

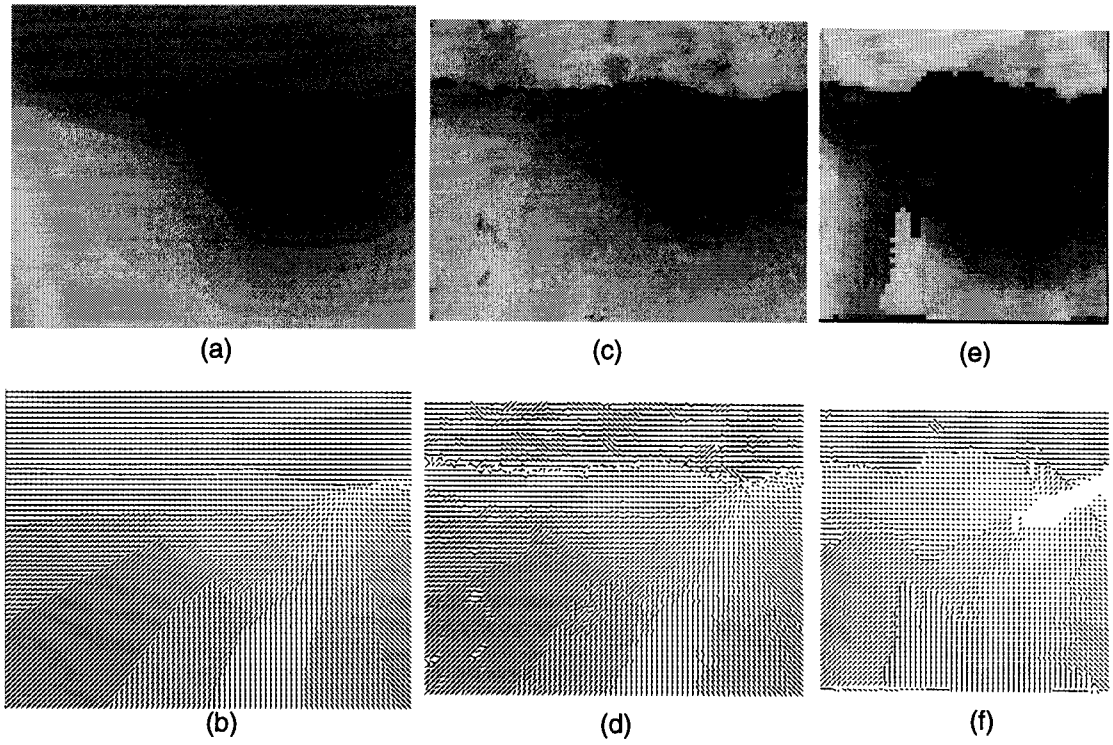


Figure 12: (a) True optical flow magnitude and (b) needlemap for frame 14 of Yosemite fly-by. (c and d), Computed optical flow using Liu's algorithm, with window size at 21, post-median filtering on, and output density at 100%. (e and f), Computed optical flow using Camus's algorithm.

Horizontal strip:

This region consists of rows 200 to 220 (Figure 11). The full optical flow, vertical component of flow, and obstacles detected for this region using the true flow are depicted in Figures 13(a), (d), (g). This provides the "ground truth." Figure 13 also shows full flow, vertical component of flow, and obstacles detected using both Liu's and Camus's flow algorithms.

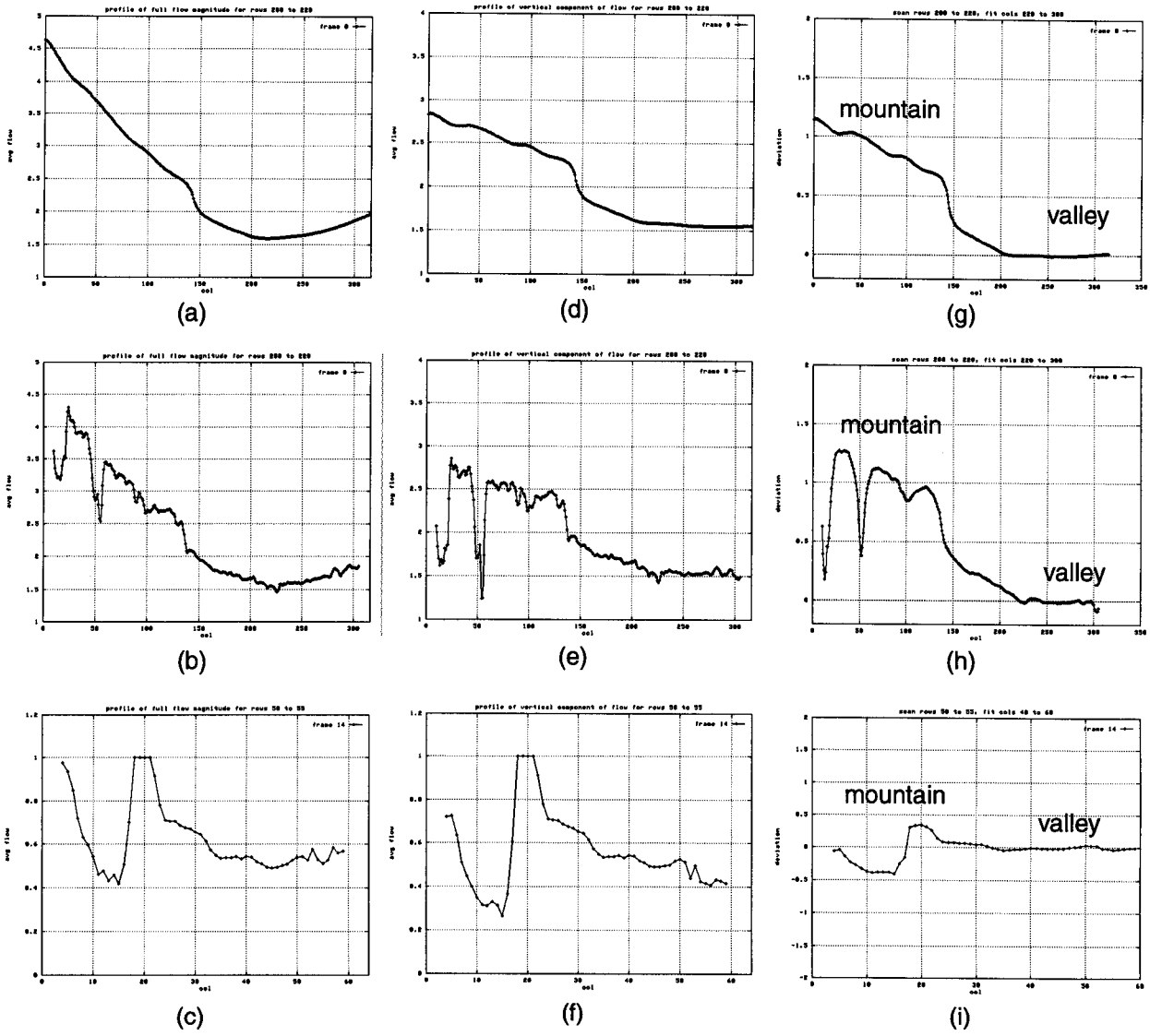


Figure 13: Horizontal strip - rows 200 to 220. Profile of full flow magnitude of (a) true optical flow, (b) Liu's flow, (c) Camus's flow. Profile of vertical component of (d) true flow, (e) Liu's flow, (f) Camus's flow. Obstacles detected using (g) true flow, (h) Liu's flow, (i) Camus's flow.

The reference line is derived from the valley region in the lower right portion of the image, and is obtained by fitting a line to the data points in columns 220 to 300 (Figure 11). The mountains can

be easily detected using both the true flow and Liu's flow. For Camus's flow, there is an error in the vertical component of flow. Where there should be a protrusion in the vertical component of flow for the mountain, there is a depression. This depression is due to the effect of block-subsampling a narrow, high contrast band and is not due to the flow algorithm itself [8] . This problem is a side effect of Camus's sampling algorithm and does not occur very often. Notice that if only the full flow profile (Figures 13(a), (b), (c)) were examined, then the rise in the profile between columns 250 and 316 could falsely have suggested an obstacle region. This rise in profile does not exist in the vertical component of flow (Figures 13(d), (e), (f)) or deviation profiles (Figures 13(g), (h), (i)).

Vertical strip:

This region consists of columns 270 to 290 (Figure 11). Similar to Figure 13, Figure 14 shows "ground truth" for the vertical strip as well as results using both Liu's and Camus's algorithms. The reference line is derived from the valley region obtained from rows 190 to 240 (Figure 11). The sky, mountain, and valley regions can be clearly differentiated in the deviation profiles (Figures 14(g), (h), (i)). The sky region accounts for the large flow magnitude errors and is apparent even in the deviation derived from the true optical flow.

For the Yosemite sequence (with 252x316 size images), the system can run up to 3.05 Hz on a Sun 20/61SX[1] using Camus's algorithm to compute the flow (Table 1). The performance is approximately doubled on a Sun Ultra 1/140. The flow computation is the most time consuming part of the system. The obstacle detection algorithm itself is fast, and can run up to 15.6 Hz on a Sun 20/61SX using 64x64 precomputed flow (Table 2).

5.4 NASA Coke Can Sequence

This is a real diverging image sequence (Figure 15). Two regions are selected for obstacle detection: an upper horizontal strip consisting of rows 45 to 90, and a lower horizontal strip consisting of rows 220 to 260. For the top strip, we would expect to see protrusions for the pencils on each side of the image and the Coke can in the middle. For the bottom strip, we would expect to see protrusions for the plate on the left and the pencil on the right.

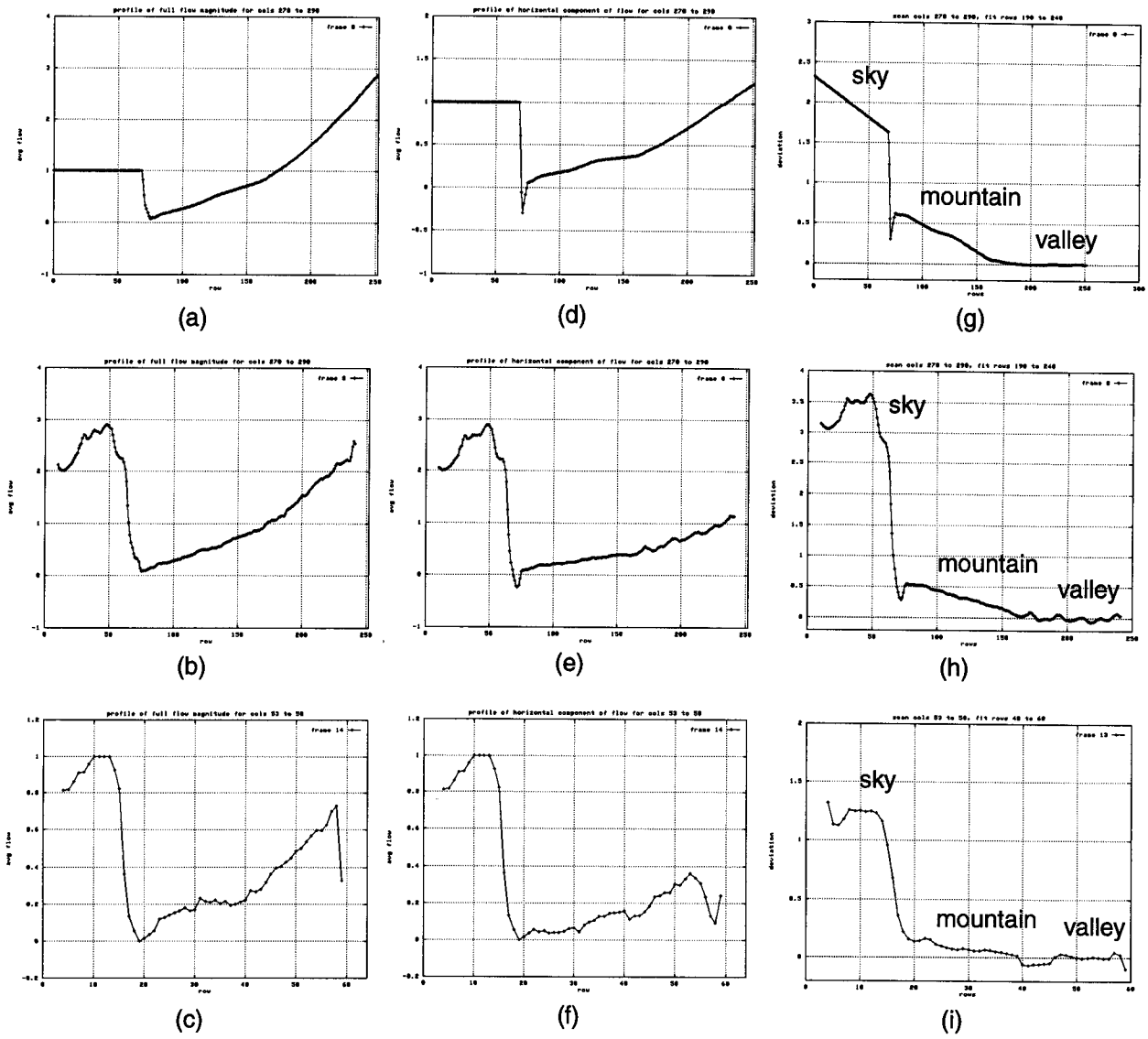


Figure 14: Vertical strip - columns 270 to 290. Profile of full flow magnitude of (a) true optical flow, (b) Liu's flow, (c) Camus's flow. Profile of horizontal component of (d) true flow, (e) Liu's flow, (f) Camus's flow. Obstacles detected using (g) true flow, (h) Liu's flow, (i) Camus's flow.

We use Liu's algorithm to compute the optical flow because his algorithm works well with small flow values. To compute the deviation for the bottom strip, the reference line is derived from the region contained within columns 110 to 210. The deviation output (Figure 16(c)) shows two protrusions on the left side, which correspond to the plate. The deviation output does not really show a protrusion for the pencil on the right because there is little flow in the vertical direction for that

1.Certain commercial equipment, instruments, or materials are identified in this paper in order to adequately specify the experimental procedure. Such identification does not imply recommendation or endorsement by NIST, nor does it imply that the materials or equipment identified are necessarily best for the purpose.

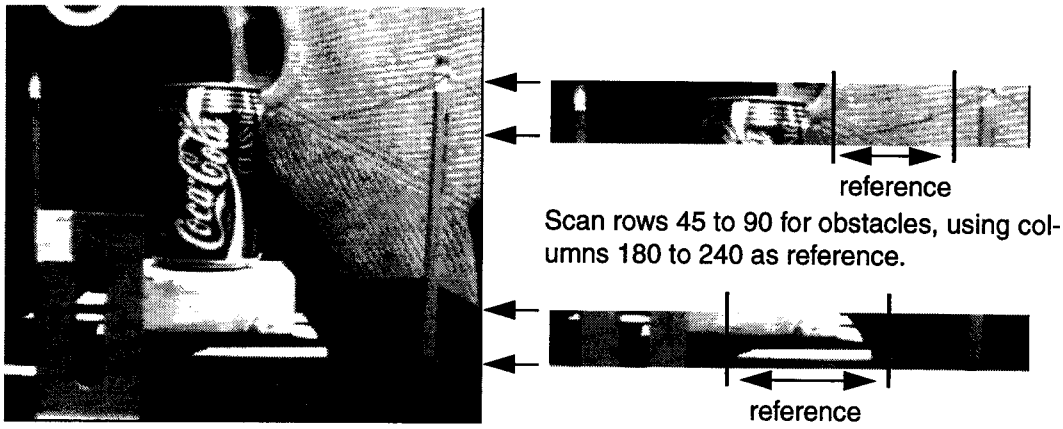


Figure 15: NASA Coke can sequence.

Table 1: Performance of system (includes subsampling, computing optical flow, and computing deviation) on a Sun 20/61SX for the Yosemite sequence (252x316 size image, 15 frames) using rows 200 to 220 as scan lines and columns 220 to 300 as reference.

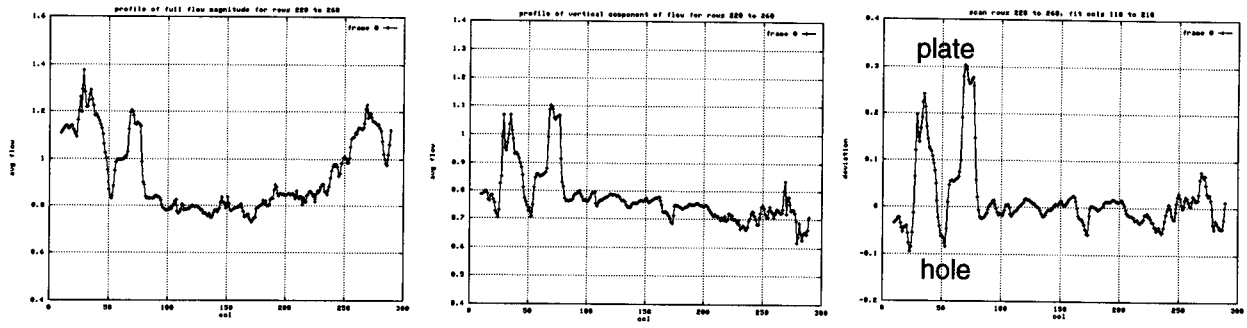
Flow algorithm	Image size	Sub-sampling factor	Flow algorithm parameters	Obstacle detection parameters	Frames per second
Camus	256x256	4	delays at 1/10, low texture thresholding off	default (reference line median filter window 3x3, deviation median filter window 3x3)	3.05
Liu	252x316	1	window size 21x21, post median filtering on, output density 100%	default	0.04968

Table 2: Performance of algorithm on a Sun 20/61SX using precomputed flow and the same obstacle detection parameters as in Table 1.

Flow algorithm	Image size	Frames per second
Camus	64x64	15.6
Liu	252x316	0.9987

region (Figure 16(b)).

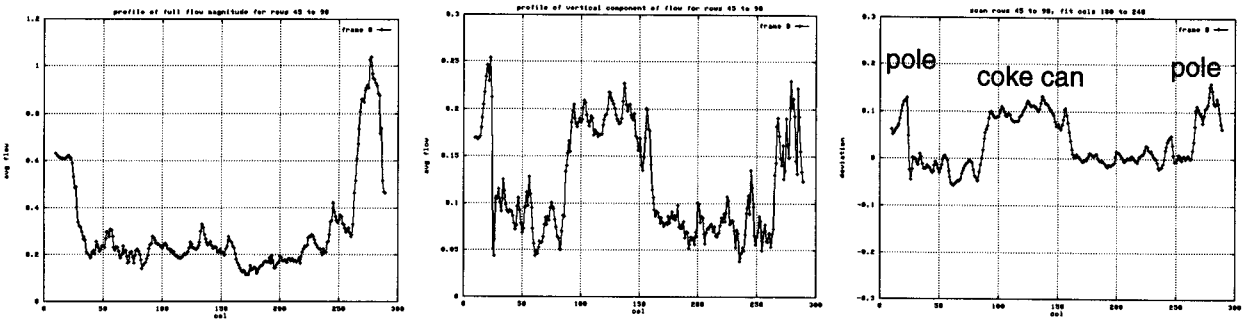
To compute the deviation for the top strip, we select columns 180 to 240 to generate the reference line. The deviation output (Figure 17(c)) shows three protrusions that correspond to the pencils (or poles) and the Coke can. Notice that if only the full flow profile (Figure 17(a)) were examined, then



(a) (b) (c)

Figure 16: Results for bottom strip of NASA sequence (rows 220 to 260). (a) Profile of full flow magnitude using Liu's flow. (b) Profile of vertical component of flow. (c) Obstacles detected using columns 110 to 210 as reference.

it would be difficult to extract the Coke can. However, the Coke can is clearly visible in the vertical component of flow (Figure 17(b)), as well as in the deviation profile.



(a) (b) (c)

Figure 17: Results for top strip of NASA sequence (rows 45 to 90). (a) Profile of full flow magnitude using Liu's flow. (b) Profile of vertical component of flow. (c) Obstacles detected using columns 180 to 240 as reference.

5.5 NIST Three-Chairs Sequence

These image sequences were taken from a moving robot inside the laboratory. (Figure 18(a) shows an example.) The objective of this experiment is to find the chairs, which act as obstacles. We select scan rows 170 to 180 to plot the deviation, using columns 60 to 90 to generate the reference line. We use the same scan and reference lines throughout the image sequence. Initially, the reference line corresponds to the ground on the left side of the image. As the image sequence advances, the reference line corresponds to the left chair. As an obstacle is approached, the magnitude of flow due to the obstacle gradually increases. This accounts for a gradual increase in deviation values, which is displayed as a gradual increase in height of the protrusion representing the obstacle.

In the deviation output for frame 41 (Figures 18(d), (e)), the middle chair is clearly visible. Using

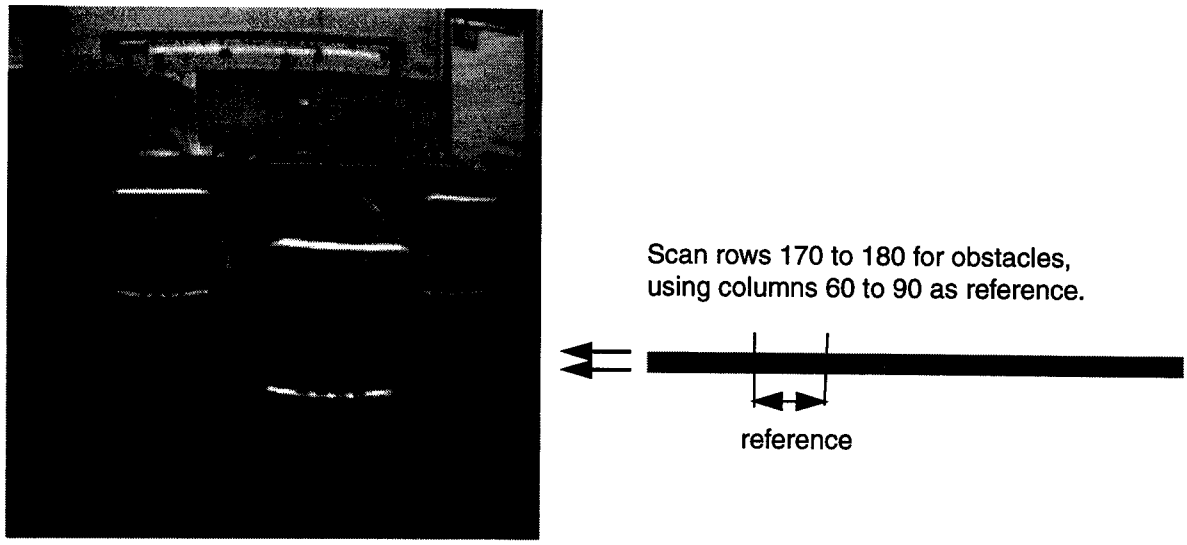
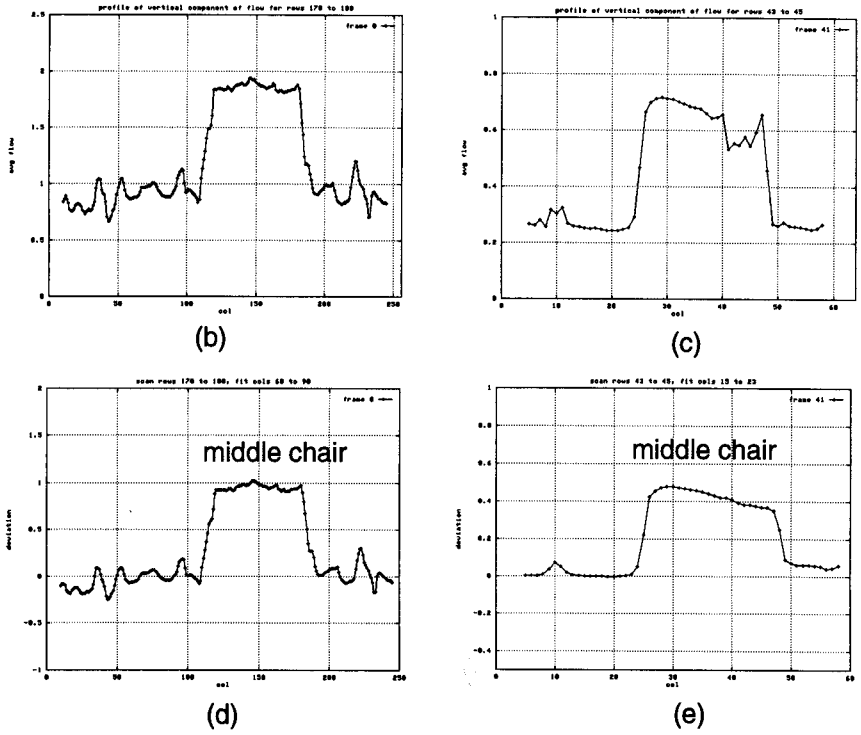


Image size: 256x256.

(a)



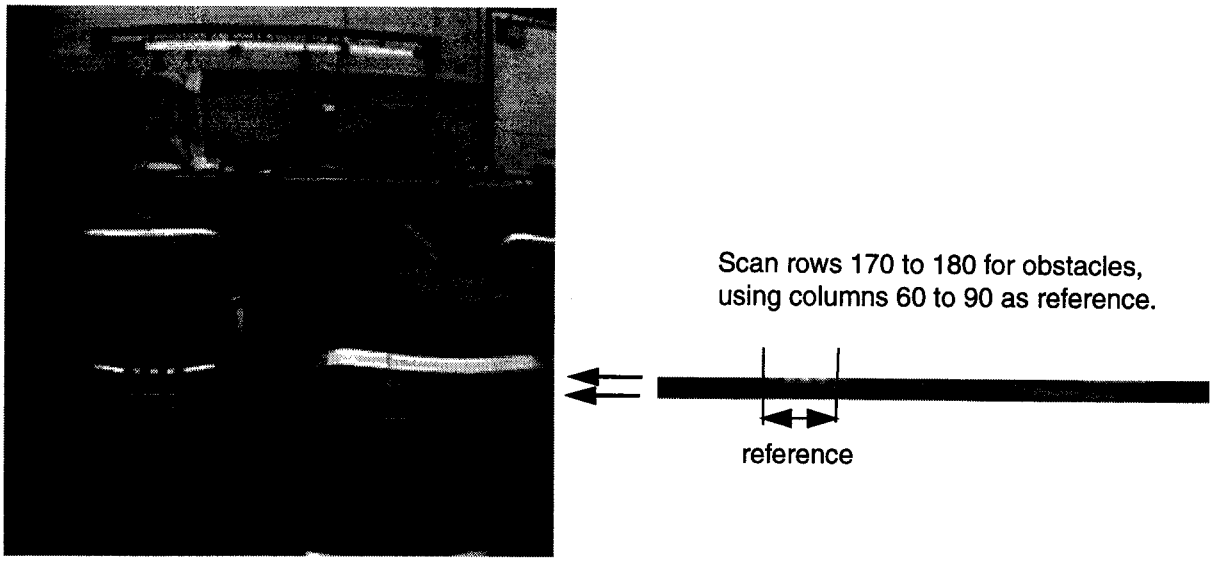
(b) (c)

(d) (e)

Figure 18: (a) NIST chair sequence - frame 41. (b) Vertical component of Liu's flow. (c) Vertical component of Camus's flow. (d) Obstacles detected with Liu's flow. (e) Obstacles detected with Camus's flow.

the same scan lines for frame 63 (Figure 19(a)), the protrusions representing the left and middle chairs can be seen (Figures 19(d), (e)). Notice in Figure 19 that the reference line is derived from points on the left chair. Thus, in the deviation output, the left chair has zero deviation (Figures 19(d), (e)). The floor has negative deviation because it has less vertical component of flow than the

left chair. Intuitively, this is correct because the left chair is closer to the camera than the floor. Likewise, the right chair has positive deviation because it is closer to the camera than the left chair.



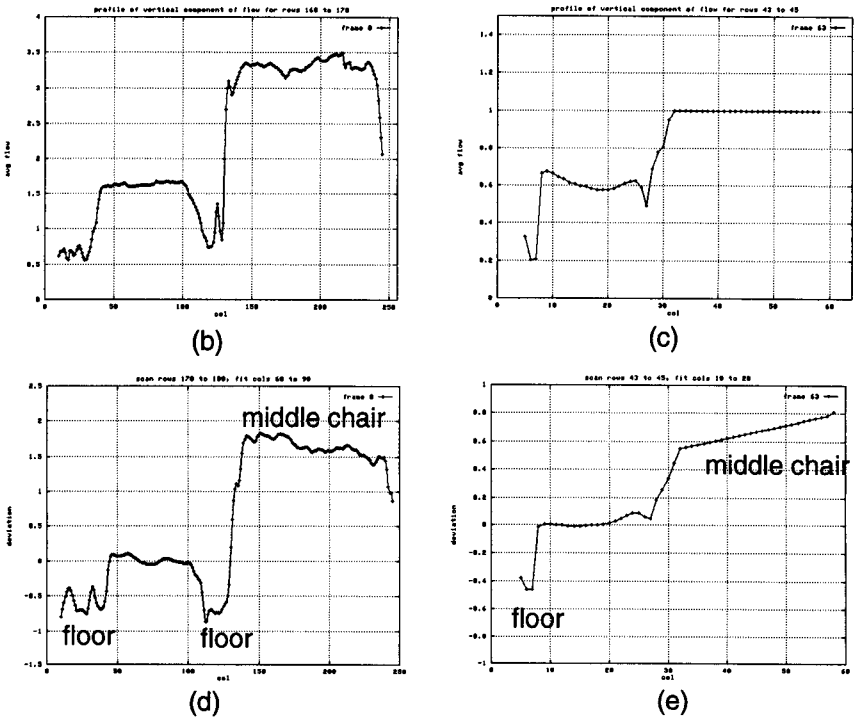


Figure 19: (a) NIST chair sequence - frame 63. (b) Vertical component of Liu's flow. (c) Vertical component of Camus's flow. (d) Obstacles detected with Liu's flow. (e) Obstacles detected with Camus' flow.

For these chair sequences (with 256x256 size images), the system ran up to 7.167 Hz on a Sun Ultra 1/140 using Camus's algorithm to compute the flow (Table 3). The obstacle detection algorithm

itself can run up to 46.4 Hz on this computer using 64x64 precomputed flow (Table 4).

Table 3: Performance of system (includes subsampling, computing optical flow, and computing deviation) on a Sun Ultra 1/140 for the chair sequence (256x256 size image, 84 frames) using rows 170 to 180 as scan lines and columns 60 to 90 as reference.

Flow algorithm	Sub-sampling factor	Flow algorithm parameters	Obstacle detection parameters	Frames per sec
Camus	4	delays at 1/10, low texture thresholding off	default (reference line median filter window 3x3, deviation median filter window 3x3)	7.167
Liu	4	window size 5x5, post median filter on, output density 100%	default	1.689
	1	window size 21x21, post median filter on, output density 100%	default	0.0627

Table 4: Performance of algorithm on a Sun Ultra 1/140 using precomputed flow and the same obstacle detection parameters as Table 3.

Flow algorithm	Image size	Frames per sec
Camus	64x64	46.4
Liu	64x64	41.4
	256x256	1.348

Figure 20(a) shows frame 25 of the chair sequence where a vertical strip consisting of columns 130 to 150 is used to detect obstacles. Rows 20 to 24, which correspond to a region on the back of the wall, are used to generate the reference line. The middle chair is easily detected using both Liu's and Camus's flow algorithms (Figures 20(f), (g)). This example again demonstrates the power of using perpendicular flow, rather than full flow to detect obstacles. The middle chair would be much more difficult to detect in the full flow profiles (Figures 20(b), (c)).

In the experiments thus far described, the flow is computed for the full image even if the deviations are computed only in a single strip. Because the flow computation takes up the most processing time, the system performance for a single strip can be improved by cropping the image to compute flow only for the strip under consideration. Figures 21 and 22 show two such examples, where the chair sequence (Figure 18(a)) is cropped to form a strip between rows 170 and 180. The chair ob-

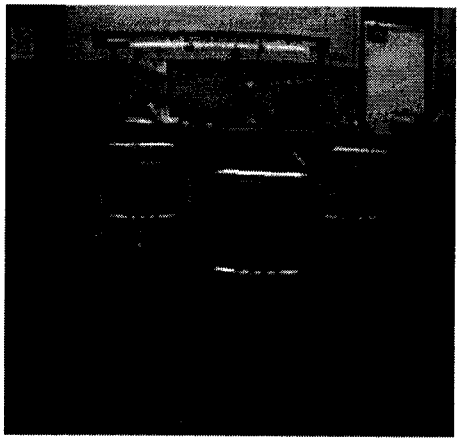


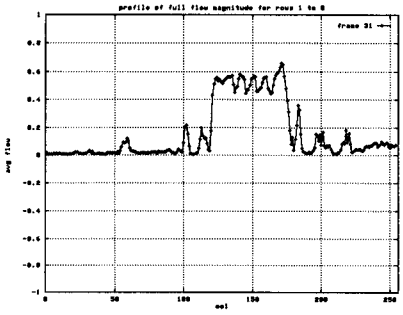
Image size: 256x256.

reference

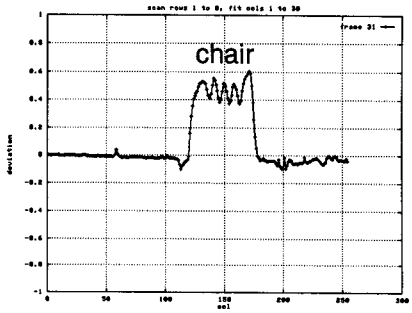
Scan columns 130 to 150 for obstacles,
using rows 20 to 44 as reference.

Figure 20: (a) NIST chair sequence - frame 25.

(a)



(b)



(c)

Figure 21: (a) NIST cropped chair sequence - frame 37, rows 170 to 180; (b) full flow magnitude using Liu's algorithm; (c) obstacles detected.

stacles are detected in both examples using Liu's flow algorithm. For these images (size 10x256), the system's performance (including computation of flow using Liu's algorithm and computation of deviation) on a Sun Ultra 1/140 is 4.04 Hz.

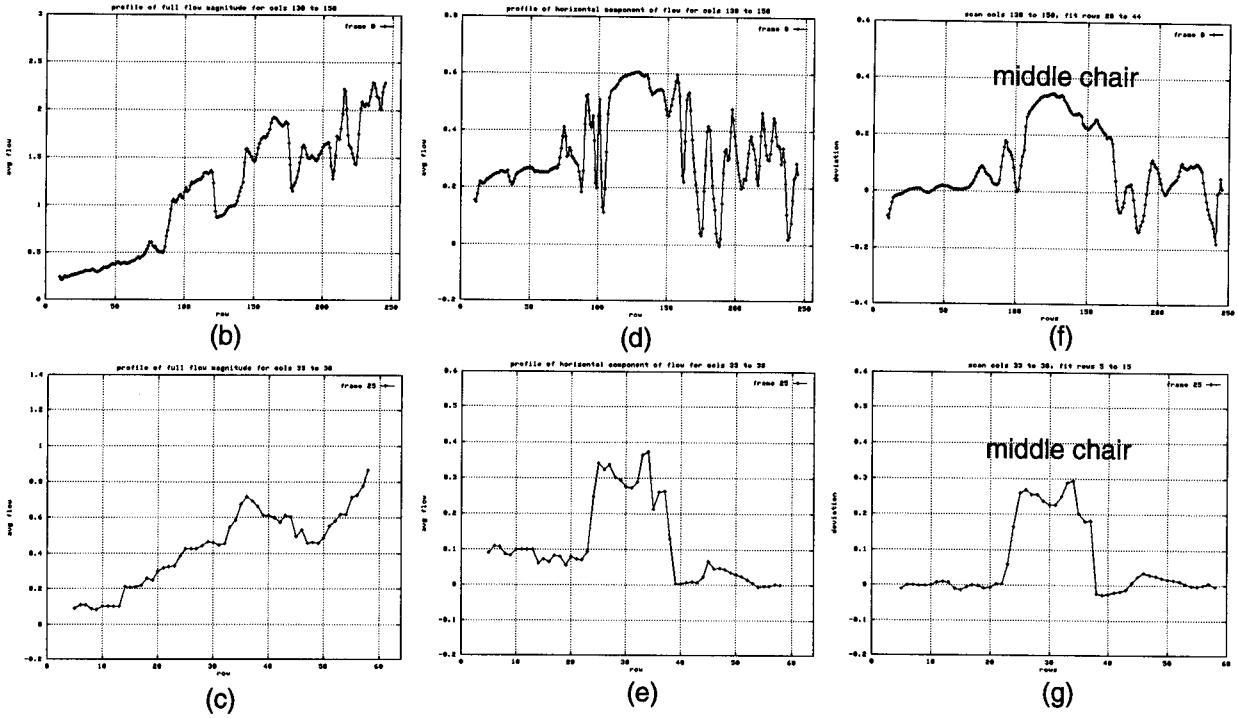


Figure 20: Full flow magnitude for (b) Liu's flow, (c) Camus's flow. Horizontal component of (d) Liu's flow, (e) Camus's flow. Obstacles detected with (f) Liu's flow, (g) Camus's flow.

(a)

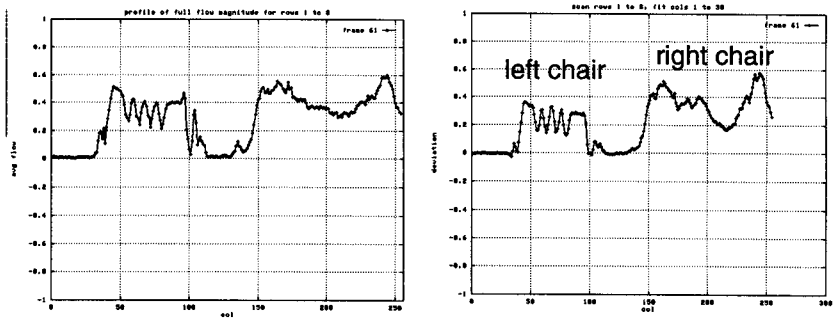


Figure 22: (a) NIST cropped chair sequence - frame 67, rows 170 to 180; (b) full flow magnitude using Liu's algorithm; (c) obstacles detected.

The application of our algorithm on a NIST laboratory sequence taken with a wide-angle camera is shown in Figure 23. The horizontal strip from rows 140 to 160 have been selected to perform obstacle detection. The reference line is generated from columns 100 to 150, which correspond to a region on the floor. The left chair, right chair, as well as rack on the right side can be detected in the deviation profile (Figure 23(b)).

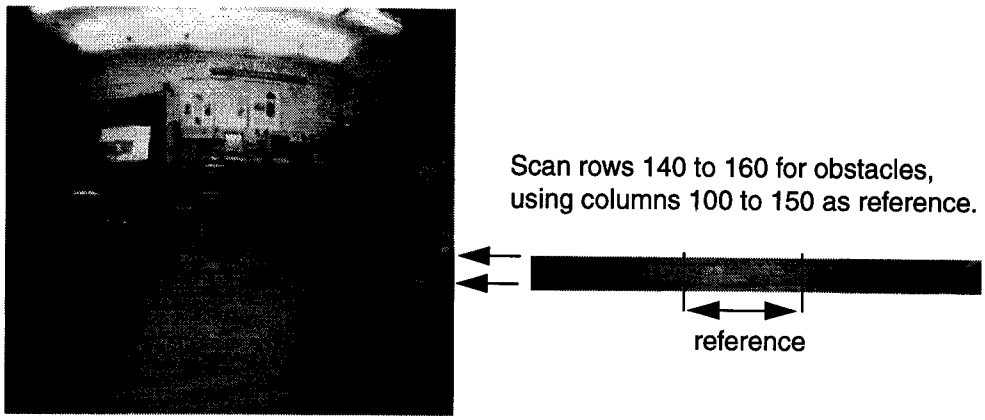
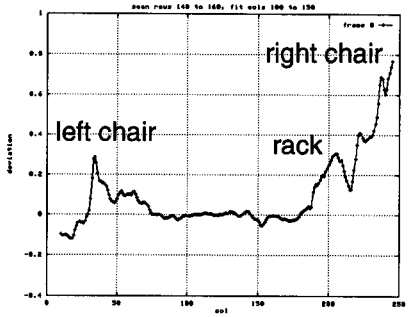


Image size: 240x256.

(a)



(b)

Figure 23: (a) NIST wide angle lab sequence; (b) obstacles detected using Liu's flow.

5.6 HMMWV Sequence

The following experiments demonstrate the use of our approach on real outdoor image sequences. These sequences are much noisier than the indoor tabletop and chair sequences because the outdoor sequences were first recorded on a VHS videotape and then digitized. The indoor sequences were digitized and stored on a disk directly from the camera.

The first outdoor experiment involves detecting a stationary car on a road. The camera is mounted on an Army HMMWV vehicle, whose motion is unknown. An image frame of the sequence is shown in Figure 24(a). Optical flow is obtained using an early version of Liu's algorithm [24] . The vertical component of flow is shown in Figure 24(b). Figure 24(c) shows the vertical component of flow for row 118. To reduce the effects of highly noisy data, a spatio-temporal median filter of size 5x5x5 is applied to the optical flow images. The reference line for row 118 is obtained by fitting a line to the data points between columns 20 and 70 at rows 117, 118 and 119. These data points, corresponding to the leftmost cluster in Figure 24(c), arise from points lying on the ground

plane. The results of obstacle detection for rows 118 and 117 are shown in Figures 24(d), (e). All the rows in the middle portion of the image were processed, and obstacle points were extracted at each row. Figure 24(f) shows all the obstacle points superimposed on the image in Figure 24(a). The white region denotes the detected obstacle

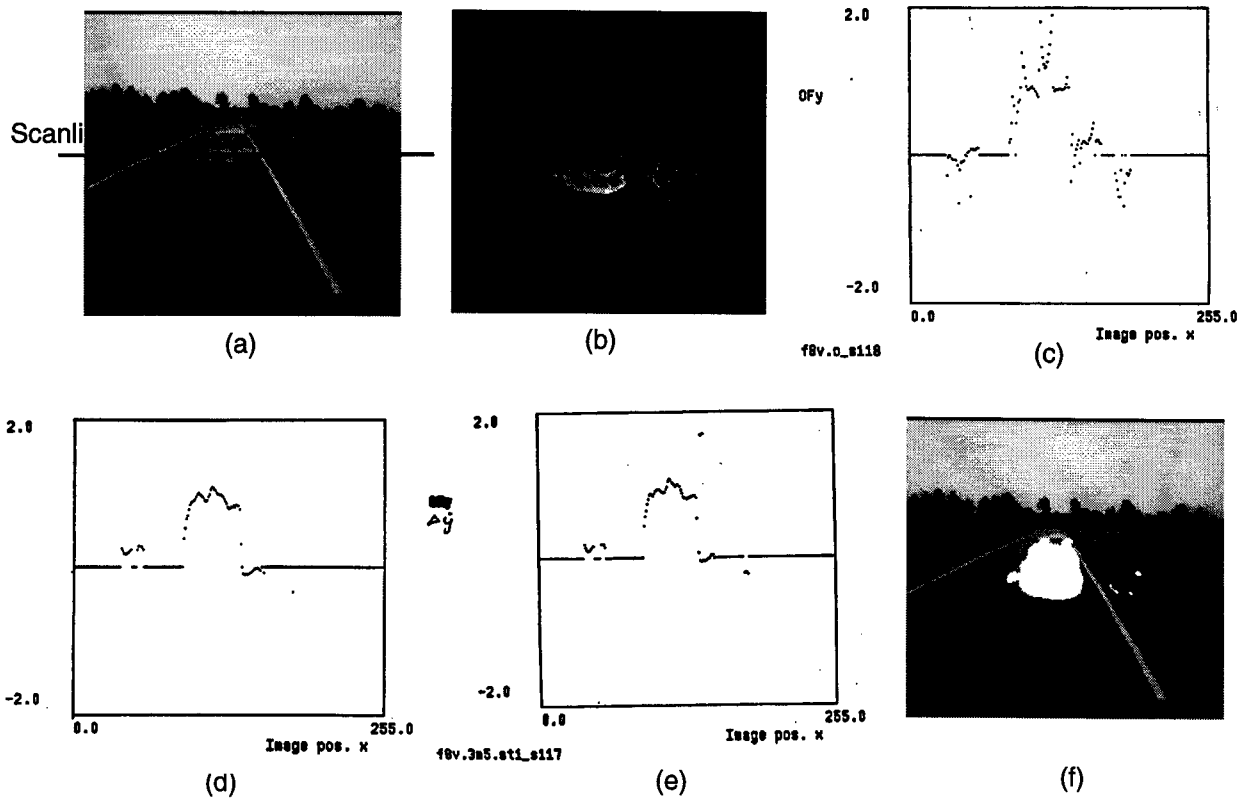


Figure 24: (a) One image frame of HMMWV sequence. (b) The vertical component of flow. (c) Vertical component of flow for row 118. (d) Obstacles detected at row 118. (e) Obstacles detected at row 117. (f) Obstacles detected for the full image.

5.7 Parking Lot Sequences

In the Metro parking lot sequence (Figure 25(a)), a camera is held by a passenger in a moving vehicle approaching several parked cars on the right side. Optical flow is obtained using the first-order, local differential method of Lucas and Kanade [29] implemented by Barron et al. [4] . Here, we only consider rows 130 and 133 labelled in Figure 25(a). The vertical components of flow in rows 130 and 133 are shown in Figures 25(b), (c). To reduce the effects of highly noise data, a median filter is applied to the optical flow image. The reference lines are generated from points in columns 20 to 90 which lie on the ground plane in the scene. They correspond to the leftmost clusters in Figures 25(b), (c). Figures 25(d), (e) show the results. The protrusions denoting parked cars on

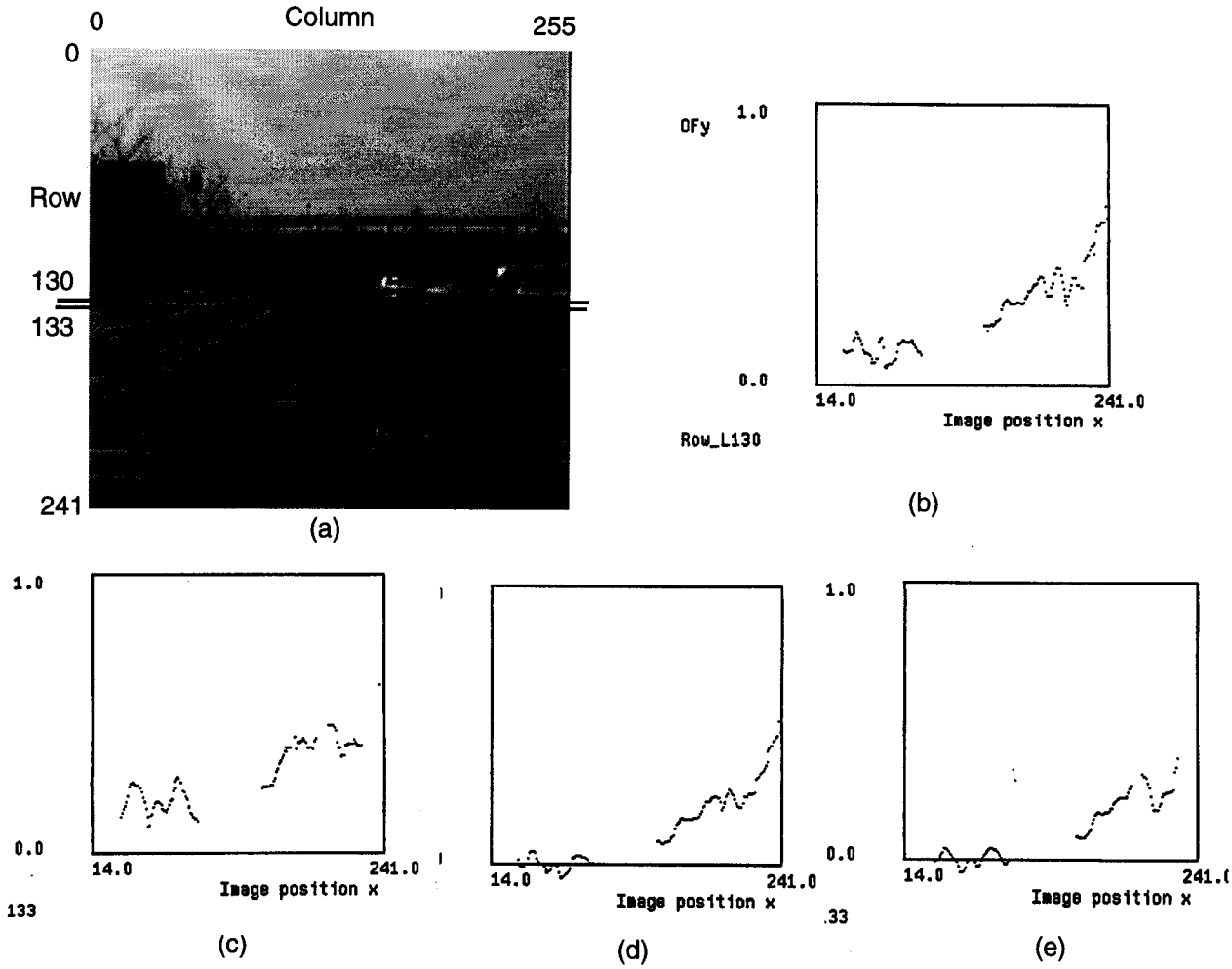


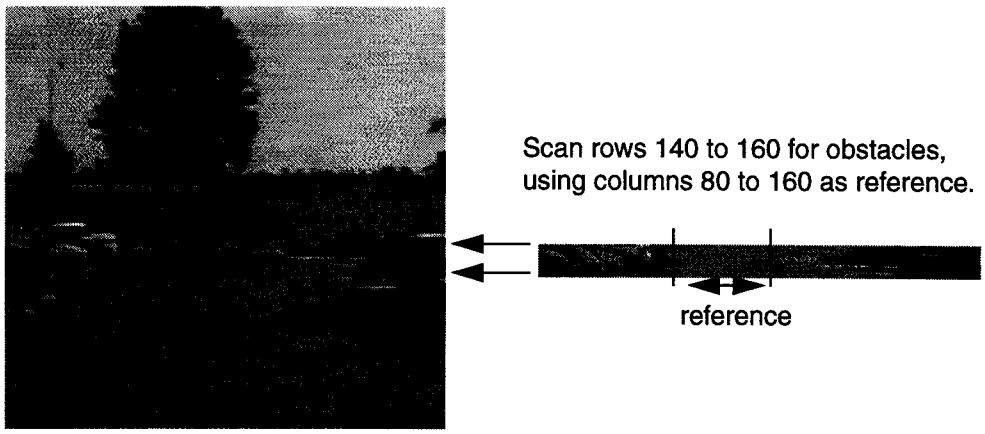
Figure 25: (a) One image frame of Metro parking lot sequence. (b) Vertical component of flow at scan line 130. (c) Vertical component of flow at scan line 133. (d) Obstacles detected at scan line 130. (e) Obstacles detected at scan line 133.

the right side are easily detected. Note that the rightmost image positions show the largest deviations. This is because the parked cars in the rightmost image positions are closest to the observer.

An image in the NIST parking lot sequence is shown in Figure 26(a). We select rows 140 to 160 to plot the deviation, using columns 80 to 160 (corresponding to points on the road) to generate the reference line. The vertical component of flow is shown in Figure 26(b); the detected obstacles are shown in Figure 26(c).

6 Conclusions

This paper shows a novel approach to obstacle detection for terrain navigation. New visual linear



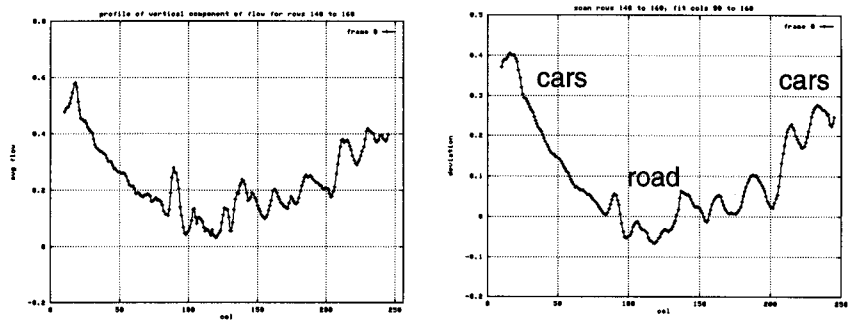


Figure 26: (a) NIST parking lot sequence. (b) Profile of vertical component of Liu's flow. (c) Obstacles detected using Liu's optical flow for rows 140 to 160.

invariants based on optical flow have been derived. They require the use of appropriate image-based spaces. Employing the linear invariance property, obstacles can be discriminated from terrain by using reference flow lines obtained from measured optical flow.

The approach has several advantages:

(1) Simple - Only one component of optical flow is needed. Knowledge about camera-to-ground coordinate transforms, or specific vehicle (or camera) motion is not required. No range, 3-D motion, or 3-D scene geometry is recovered; 2-D visual information (i.e., optical flow) is directly used to detect obstacles.

(2) General - The method is valid for a vehicle (or camera) moving under general motion, i.e., arbitrary translation and rotation.

(3) Fast - The method is simple and fast. We have also demonstrated that the method can be suc-

cessfully used with fast optical flow algorithms.

(4) Minimal Error Sources - The error sources involved are reduced to a minimum because the only required information is one component of the optical flow.

Experiments with both synthetic and real image data suggest that the approach using visual invariants as a tool for obstacle detection is effective. The method has been demonstrated with both ground and air vehicle scenarios.

To use the method presented here in a real-time robotic obstacle avoidance system, several issues must still be addressed. The first is automatically extracting obstacles from the deviation profiles. The deviation represents the difference in flow arising from points on an obstacle and flow arising from points on a reference region (which is often a terrain region of interest; Figure 3). The magnitude of the deviation is a function of the difference in depth between points on the obstacle and points in the reference region. The extent to which this deviation can be detected determines the extent to which the obstacles can be detected. Therefore, the deviation must significantly exceed the noise in the flow.

Consider the case of objects of different size lying on the ground at the same distance from the camera. Assume the camera is moving forward and the y component of flow along a horizontal strip is used to measure deviation. For a tall obstacle, suppose that we measured flow along two horizontal strips, one near the top of the obstacle and one near the bottom. The difference in depth between points on the obstacle and points on the ground is larger for the top strip, and therefore the flow deviation would be larger as well. Therefore, the ratio of deviation to noise will be higher for the top strip, making the obstacle more detectable if the top strip is used. For this reason, tall obstacles will tend to be more detectable than short ones. The size of the smallest obstacle detectable depends critically on the accuracy and noise characteristics of the flow algorithm being used. To automatically extract obstacles, methods based on this type of analysis must be developed for analyzing the deviation profiles.

A second issue is automatically choosing the points in the perpendicular flow to generate reference flow lines. Earlier, we described that one heuristic involves choosing points near the bottom of the

image, because these can often be assumed to correspond to the ground surface near the vehicle. Other approaches need to be investigated in the future.

This paper has addressed the problem of obstacle detection, rather than obstacle avoidance. A third issue is therefore how to use our detection method for obstacle avoidance. Currently, our method indicates (a) the presence of an obstacle and (b) the position of that obstacle in the image. The distance to the obstacle is a 3-D quantity that is often used for avoidance maneuvers. However, it has been demonstrated that obstacle avoidance can be performed using time-to-collision, which can be computed directly from optical flow or even from temporal changes in image gradients without 3-D reconstruction [3] [9] [10] [11] [26] [33] [37] . We propose in the future to apply such time-to-collision approaches to the detected obstacles.

The accuracy and usefulness of the method presented here depends on the accuracy of the estimated optical flow. We have demonstrated results using several flow algorithms. One of the fastest is Camus's algorithm, while one of the most accurate is Liu's algorithm [25] . The optimal manner in which to combine such algorithms in an integrated system (e.g., use Camus's fast algorithm for nearby obstacles with large flows and quick response-time requirements, while use Liu's slower algorithm for distant obstacles with small flows and longer response-time requirements) needs to be studied in the future.

A general difficulty with optical flow-based approaches for navigation is that camera jitter due to driving over rough terrain makes it difficult to accurately estimate flow. Camera or image stabilization approaches (either mechanical or electronic) [6] [45] are, therefore, required for flow-based navigation.

All of the issues described above must be resolved to develop a real-time integrated obstacle avoidance system based on the methods described in this paper.

7 Acknowledgements

The authors thank James Albus and David Coombs for important comments on this work. They also thank Herbert Lau for his experiments and software for extracting optical flow, Marilyn Nash-

man for her software that superimposes two images, and Ted Camus and Hongche Liu for providing us their optical flow software. Special thanks are due to J. L. Barron for his software that implements various optical flow algorithms.

References

[1] J. S. Albus, "Outline for a Theory of Intelligence," *IEEE Transactions on System, Man, and Cybernetics*, Vol. 21, No. 3, 1991.

[2] J. S. Albus and T.-H. Hong, "Motion, Depth, and Image Flow," *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1990.

[3] Y. Aloimonos, "Is Visual Reconstruction Necessary? Obstacle Avoidance Without Passive Ranging," *Journal of Robotic Systems* 9(6), 843-858, 1992.

[4] J. Barron, D. Fleet, and S. Beauchemin, "Performance of Optical Flow Techniques," International Journal of Computer Vision, Vol. 12, No. 1, 43-77, 1994.

[5] B. Bhanu, B. Roberts, and J. Ming, "Inertial Navigation Sensor Intergated Motion Analysis for Obstacle Detection," *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1990.

[6] P. Burt and P. Anandan, "Image Stabilization by Registration to a Reference Mosaic," *Proc. ARPA Image Understanding Workshop*, 1994.

[7] T. Camus, "Real-Time Quantized Optical Flow," *Proc. IEEE Conf. on Computer Architectures for Machine Perception*, Como, Italy, 1995.

[8] T. Camus, "Real-Time Quantized Optical Flow," *Real-Time Imaging*, in press.

[9] T. Camus, D. Coombs, M. Herman, and T.-H. Hong, "Real-time Single-workstation Obstacle Avoidance Using Only Wide-field Flow Divergence," *Proc. 13th International Conference on Pattern Recognition*, 1996.

[10] D. Coombs, M. Herman, T.-H. Hong, and M. Nashman, "Real-Time Obstacle Avoidance Using Central Flow Divergence and Peripheral Flow," *Proc. Fifth International Conference on Computer Vision*, pp. 226-283, 1995.

[11] D. Coombs, M. Herman, T. -H. Hong, and M. Nashman, "Real-Time Obstacle Avoidance Using Central Flow Divergence and Peripheral Flow," *IEEE Transactions on Robotics and Automation*, in press.

[12] M. J. Daily, J. G. Harris, and K. Reiser, "Detecting Obstacles in Range Imagery," *Proc. DARPA Image Understanding Workshop*, 1987.

[13] R. T. Dunlay and D. G. Morgenthaler, "Obstacle avoidance on roadways using range data," *SPIE Mobile Robots*, Vol. 727, 1986.

[14] W. Enkelmann, "Obstacle Detection by Evaluation of Optical Flow Fields from Image Sequences," *First European Conf. on Computer Vision*, 1990.

[15] J. J. Gibson, *The Ecological Approach to Visual Perception*, Lawrence Erlbaum Associates, 1986.

[16] M. Hebert, T. Kanade, and I. Kweon, "3-D Vision Techniques for Autonomous Vehicles," *In Analysis and Interpretation of Range Images*, R. C. Jain and A. K. Jain, eds., Springer-Verlag, 1990.

[17] D. Heeger and A. Jepson, "Subspace Methods for Recovering Rigid Motion I: Algorithm and Implementation," *International Journal of Computer Vision*, 7:2, 95-117, 1992.

[18] M. Herman and T.-H. Hong, "Visual Navigation using Optical Flow," *Proc. NATO Defense Research Group Seminar on Robotics in the Battlefield*, Paris, France, March 1991.

[19] M. Herman, M. Nashman, T.-H. Hong, H. Schneiderman, D. Coombs, G.-S. Young, D. Raviv, and A. J. Wavering, "Minimalist Vision for Navigation," In *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*, Aloimonos, Y., ed., Lawrence Erlbaum Associates: Mahwah, NJ, 1997.

[20] M. Herman, D. Raviv, H. Schneiderman, and M. Nashman, "Visual Road Following Without 3-D Reconstruction," *Proc. SPIE 22nd Applied Imagery Pattern Recognition Workshop*, Vol. 2103, 1993.

[21] W. Hoff and C. Sklair, "Planetary Terminal Descent Hazard Avoidance Using Optical Flow ," *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1990.

[22] H. Lau, "Optical Flow Extraction with Motion Known A Priori," University of Maryland, Master's Thesis, Electrical Engineering Department, May 1992.

[23] R. Lenz and R. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology," *IEEE Transactions on Parttern Analysis and Machine Intelligence,* Vol. 10, No. 5, 1988.

[24] H. Liu, T.-H. Hong, M. Herman, and R. Chellappa, "A Reliable Optical Flow Algorithm Using 3-D Hermite Polynomials," NISTIR-5333, NIST, Gaithersburg, MD, December 1993.

[25] H. Liu, T.-H. Hong, M. Herman, and R. Chellapa, "Accuracy vs. Efficiency Trade-offs in Optical

Flow Algorithms," *Proc. Fourth European Conference on Computer Vision*, 1996.

[26] H. Liu, T.-H. Hong, M. Herman, and R. Chellapa, "Image Gradient Evolution - A Visual Cue for Collision Avoidance," *Proc. 13th International Conference on Pattern Recognition*, 1996.

[27] H. Liu, T.-H. Hong, M. Herman, and R. Chellapa, "A General Motion Model and Spatio-temporal Filters for Computing Optical Flow," NISTIR-5539, NIST, Gaithersburg, MD, November 1994; *International Journal of Computer Vision*, in press.

[28] H. C. Longuet-Higgins and K. Prazdny, "The Interpretation of a Moving Retinal Image," *Proc. R. Soc. Lond. B208*, pp. 385-397, 1980.

[29] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proc. DARPA Image Understanding Workshop*, pp. 121-130, 1981.

[30] H. A. Mallot et al., "Inverse Perspective Mapping Simplifies Optical Flow Computation and Obstacle Detection," *Biological Cybernetics*, Vol. 64, pp177-185, 1991.

[31] L. Matthies, "Toward Stochastic Modeling of Obstacle Detectability in Passive Stereo Range Imagery," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 1992.*

[32] K. Nakayama, "Biological Image Motion Processing: A Review," *Vision Research,* Vol. 25, No. 5, pp. 625-660, 1985.

[33] R. C. Nelson and J. Aloimonos, "Obstacle Avoidance using Flow Field Divergence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 10, 1989.

[34] D. N. Oskard, T. H. Hong, and C. A. Shaffer, "Real-Time Algorithms and Data Structures for Underwater Mapping," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 20, No. 6, 1990.

[35] D. Raviv, "Flat Surfaces: A Visual Invariant," *NISTIR 4794*, NIST, Gaithersburg, MD, Mar. 1992.

[36] D. Raviv and M. Herman, "Visual Servoing from 2-D Image Cues," In *Active Perception*, Y. Aloimonos, ed., Lawrence Erlbaum Associates, Hillsdale, NJ, 1993.

[37] G. Sandini, F. Gandolfo, E. Grosso, and M. Tistarelli, "Vision During Action," In *Active Perception*, Y. Aloimonos, ed., Lawrence Erlbaum Associates, Hillsdale, NJ, 1993.

[38] S. Singh and P. Keller, "Obstacle Detection for High Speed Autonomous Navigation," *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1991.

[39] U. Solder and V. Graefe, "Object Detection in Real Time," *Proc. SPIE, Vol. 1388 Mobile Robots V*, 1990.

[40] B. Sridhar, R. Suorsa and P. Smith, "Vision Based Techniques for Rotorcraft Low Altitude Flight," *Proc. SPIE* , Vol. 1571, 1991.

[41] K. Storjohann, T. Zielke, H. A. Mallot and W. Seelen, "Visual Obstacle Detection for Automatically Guided Vehicles," *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1990.

[42] M. Tistarelli and G. Sandini, "Dynamic Aspects in Active Vision," *Computer Vision, Graphics, and Image Processing: Image Understanding*, Vol. 56, No. 1, 1992.

[43] C. Tomasi and T. Kanade, "Shape and Motion from Image Streams under Orthography: A Factorization Method," *International Journal of Computer Vision*, Vol. 9, No. 2, pp. 137-154, 1992.

[44] P. A. Veatch and L. S. Davis, "Efficient Algorithms for Obstacle Detection Using Range Data," *Computer Vision, Graphics and Image Processing*, Vol. 50, No. 1, 1990.

[45] Y. S. Yao, P. Burlina, and R. Chellapa, "Stabilization of Images Acquired by Unmanned Ground Vehicles," *Proc. ARPA Image Understanding Workshop*, 1996.

[46] G.-S. Young, "Safe Navigation and Active Vision for Autonomous Vehicles: A Purposive and Direct Solution," University of Maryland, Ph.D. Dissertation, Dept. of Mechanical Engineering, May 1993.