Technical Document 2918
July 1996

Obtaining Range from Visual Motion Using Local Image Derivatives

Hoa G. Nguyen

Naval Command, Control and
Ocean Surveillance Center
RDT&E Division

San Diego, CA
92152-5001

INTRODUCTION

In many robotic and remote operation applications, depth (or range) information at various points in the scene is required. These include autonomous navigation, landing site selection, scene reconstruction, or postprocessing interpretation of video footage.

Theoretically, depth, motion, and optical flow (generated by the relative movement between the camera and scene) are three parameters of a circular problem:

- Given the camera motion and three-dimensional structure of the scene, we can generate the optical flow, and hence construct a sequence of images of the moving scene (3-D simulation).
- Given knowledge of the three-dimensional scene and the optical flow, we can compute the motion which generated it (motion recovery).
- And finally, given the camera motion and the resulting optical flow, we can extract depth information and reconstruct the three-dimensional scene that gave rise to the flow (scene reconstruction).

When only the optical flow is available, the problem becomes much harder, and the exact depth and motion cannot be determined [Horn, 1986]. Only the relative depth between various scene points, and the relative motion (or direction) can be recovered. Recovering depth and motion from a sequence of images only is an active research area. For example, Horn [1986] described a least-squares method wherein an iterative process may be used to solve a set of seven simultaneous equations involving the optical flow. Fermuller presented a tracking technique [1991] and a pattern-matching technique [in Aloimonos, 1993] to estimate motion parameters. This general problem is outside the scope of this report.

We concentrate on finding the depth given a sequence of images and known motion or direction of motion. This problem is appropriate for many real-world applications, where robot (and camera) motion can be dictated by open-loop control, or motion information can be supplied by non-visual means, such as wheel encoders, accelerometers, gyroscopes, or other non-visual feedback control schemes. Albus [1990] computed range given known motion under various conditions using the optical flow. However, the optical flow itself is difficult to compute from a sequence of images (only a component of it, the *normal flow*, is easily computable). We will show how to obtain range data without having to find the optical flow itself, and analyze the method's sensitivity to inaccuracies in the known motion. Since the method uses only temporal and spatial first derivatives, which can be computed easily from any two consecutive frames, the depth map can be computed quickly in one pass, and thus is more suitable for real-time navigational problems. Finally, we describe the result of this method when applied to a well-known sequence of test images.

BACKGROUND

We will first discuss several concepts necessary for the development of the algorithm: namely, the difference between the optical flow and the normal flow, camera geometry, computation of the normal flow, and the focus-of-expansion.

OPTICAL FLOW AND NORMAL FLOW

As the camera moves in a static environment or as an object moves in front of the camera, relative motion occurs between the camera and the objects. The *motion field* assigns velocity vectors to points in the camera image. These vectors are projections of the corresponding real-world motion vectors. On the other hand, the *optic flow* is the apparent motion of the image pattern, which is not necessarily the same as the motion field. Consider a fixed object being illuminated by a moving light source. The motion field is zero since the object is stationary. However, the optic flow is non-zero, since the brightness pattern in the image changes. Except for a few selected scenarios, we expect the optic flow to be the same as the motion field. This assumption is used by researchers in deriving useful information about the scene from visual motion. Figure 1 illustrates the calculated optical flow generated by a spinning sphere [Horn, 1986].

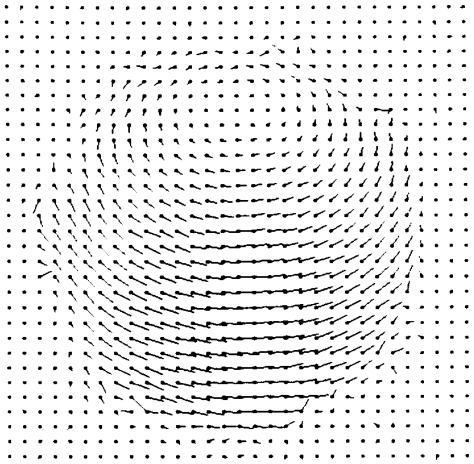

Figure 1. The optical flow computed by an iterative algorithm on simulated data of a spinning sphere on a randomly patterned background [Horn, 1986]. Note that erroneous vectors sometimes occur at boundaries, where the brightness is discontinuous.

Many vision algorithms depend on the assumption that the optic flow is available to the processing system. However, when camera motion is not known, accurate optic flow is usually not available due to a phenomenon known as the *aperture problem*, as demonstrated in figure 2. As a line of constant brightness moves across the image, which vector represents the correct optic flow at point P? That is, to which position (Q_1, Q_2, ...) has point P moved? This illustrates the fact that the optic flow is not uniquely determined by local computations. It is often estimated by interpolating between locations where it is available (such as brightness corners or specific scene features), by making assumptions on its smoothness (which is often incorrect) or using coarse-to-fine region-based techniques [Barron et al., 1992], or by iterative solutions [Ballard & Brown, 1982]. The *normal flow*, however, is unique and always simple to derive. It is the component of the optical flow perpendicular to the brightness contour (i.e., along the brightness gradient--PQ_2 in figure 2). We will give the derivation of the normal flow after we introduce the camera geometry in the next section.

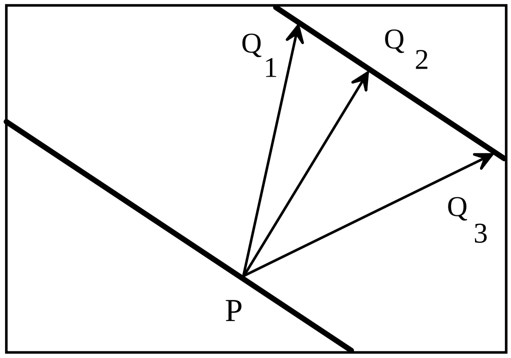

Figure 2. The aperture problem.

CAMERA GEOMETRY

We define the geometry of the camera and scene as in figure 3. The image plane is placed at a focal length *f* from the lens *O* along the optical axis, which is our Cartesian Z axis. Technically, the image plane appears on the other side of the lens. However, we have placed the image plane on the same side with the scene for convenience (the input devices normally invert the projected image so that in effect the output image appears as depicted).

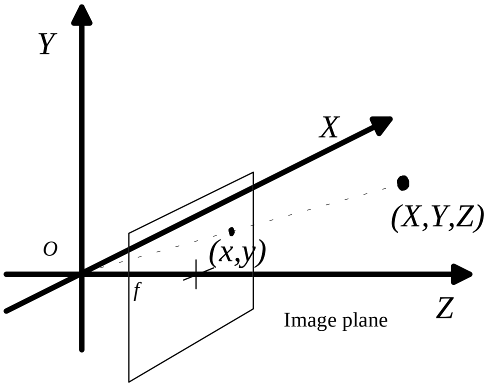

Figure 3. Perspective projection of camera and scene.

From the perspective projection, we can see that any point *(X,Y,Z)* in the scene is projected on to the image plane at *(x,y)* where:

$$x = \frac{Xf}{Z} \qquad\qquad y = \frac{Yf}{Z}$$

COMPUTATION OF THE NORMAL FLOW

Let *E(x,y,t)* be the image brightness at time *t* at image point *(x,y).* After the motion has occurred, the same image brightness will appear at point *(x+δx, y + δy)* at time *t +δt.* Thus,

$$E(x+\delta x, y+\delta y, t+\delta t) = E(x,y,t).$$

Assuming that image brightness varies smoothly with *x, y,* and *t*, we can use the Taylor series expansion on the left-hand side to get

$$E(x,y,t) + \delta x\frac{\partial E}{\partial x} + \delta y\frac{\partial E}{\partial y} + \delta t\frac{\partial E}{\partial t} + \text{high-order terms} = E(x,y,t).$$

Canceling *E(x,y,t)*, dividing both sides by δ*t*, and taking the limit as δ*t* approaches 0, the higher order terms drop out and we are left with

$$\frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t} = 0.$$

With *u(x,y) = dx/dt* and *v(x,y) = dy/dt* defined as the components of the optical flow along the *X* and *Y* axes, we have the well-known *optical flow constraint equation*

$$\frac{\partial E}{\partial x}u + \frac{\partial E}{\partial y}v + \frac{\partial E}{\partial t} = 0,$$

which can also be expressed as a dot product:

$$\left(\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}\right) \bullet (u,v) = -\frac{\partial E}{\partial t}.$$

Since the brightness gradient is $\left(\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}\right)$ and the optical flow is *(u,v),* the *normal flow* (the component of the optical flow in the direction of the brightness gradient) is

$$\frac{\left(\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}\right)}{\sqrt{\left(\frac{\partial E}{\partial x}\right)^2 + \left(\frac{\partial E}{\partial y}\right)^2}} \bullet (u,v) = \frac{-\frac{\partial E}{\partial t}}{\sqrt{\left(\frac{\partial E}{\partial x}\right)^2 + \left(\frac{\partial E}{\partial y}\right)^2}} = U_n, \qquad (1)$$

with the minus sign reflecting the fact that the normal flow is in the opposite direction to the gradient vector whenever $\frac{\partial E}{\partial t}$ is positive (e.g., at the leading edge of a bright object), and vice versa (the gradient vectors point toward brighter areas).

Note that all computations in deriving the normal flow involve only local derivatives and do not require advanced knowledge of object or camera motion. It is the only representation of image motion that can be robustly computed [Aloimonos, 1990].

THE FOCUS OF EXPANSION

As the camera moves relative to the static environment, or as an object in the scene moves with respect to the camera, the translational components of the optic flow converge at a point on the image plane, the *focus of expansion* (FOE)--see figure 4. The FOE is very useful in navigation problems because it is the projected image of the ray along which a camera undergoing translational motion moves. If the FOE falls inside an object, that object will collide with the camera.

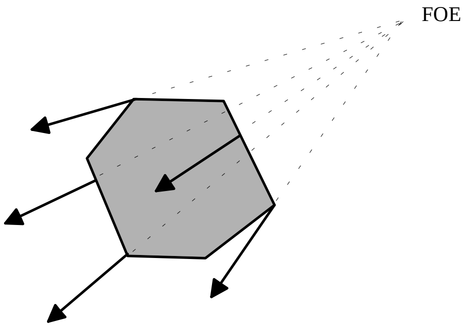

Figure 4. The FOE from an optical flow map.

If we use the camera geometry of figure 3, a rigid object moving with translational velocities *(U,V,W)* (with no rotation) that was at *(X, Y, Z)* initially will be imaged at *(x', y')* at time *t*, where

$$\left(x',y'\right)=\left(\frac{(X+Ut)f}{Z+Wt},\frac{(Y+Vt)f}{Z+Wt}\right).$$

Since the FOE is the image of the point at *t* = minus infinity, we let *t* go to $-\infty$ and obtain

$$FOE=\left(\frac{Uf}{W},\frac{Vf}{W}\right)$$

on the image plane.

If a rotational velocity is involved, then the FOE is tied to the center of rotation. This is because rotation about any arbitrary center can be expressed as rotation about another center plus a compensating translation. The motion of any point *(X,Y,Z)* on an object undergoing translational velocities *(U,V,W)* and rotational velocities *(A,B,C)* around a center of rotation (X_0,Y_0,Z_0) can be expressed as:

$$\begin{pmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{pmatrix} = \begin{pmatrix} U \\ V \\ W \end{pmatrix} + \begin{pmatrix} A \\ B \\ C \end{pmatrix} \times \begin{pmatrix} X - X_0 \\ Y - Y_0 \\ Z - Z_0 \end{pmatrix}.$$

Now if we introduce another arbitrary point, *(X_1,Y_1,Z_1),* we can rewrite the above equation as

$$\begin{pmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{pmatrix} = \begin{pmatrix} U \\ V \\ W \end{pmatrix} + \begin{pmatrix} A \\ B \\ C \end{pmatrix} \times \begin{pmatrix} X - X_1 \\ Y - Y_1 \\ Z - Z_1 \end{pmatrix} + \begin{pmatrix} X_0 - X_1 \\ Y_0 - Y_1 \\ Z_0 - Z_1 \end{pmatrix} \times \begin{pmatrix} A \\ B \\ C \end{pmatrix},$$

which is an expression of motion of the point *(X,Y,Z)* around the new center of rotation *(X_1,Y_1,Z_1).* The second cross product, which does not involve the variables *(X,Y,Z),* is the compensating translation for the entire object.

Figure 5 demonstrates this property by showing the same optical flow vectors decomposed into two different sets of translational and rotational flows corresponding to two different centers of rotation, and their appropriate FOEs. The FOE is most useful under purely translational motion or when the rotational component is known.

RANGE DERIVATION

DERIVING RANGE WITH KNOWN CAMERA MOTION

Often in mobile robotics applications, estimates of the robot's motion are available from non-visual sources. When camera motion is known, the problem of determining distances to objects in the environment is much simplified. The full optical flow is not required, but only the normal flow (or equivalently, the local derivatives), which can be robustly computed.

Computing Range Using Local Image Derivatives

Let the translational velocities of the camera be *(U,V,W)* and the rotational velocities be *(A,B,C)* with respect to the origin. Using the same coordinate system as stated previously, we can express the velocity of any point *(X,Y,Z)* on a moving object as

$$\left(\dot{X}, \dot{Y}, \dot{Z}\right) = -(U, V, W) - (A, B, C) \times (X, Y, Z),$$

which can be rewritten as

$$\begin{aligned} \dot{X} &= -U - BZ + CY, \\ \dot{Y} &= -V - CX + AZ, \\ \dot{Z} &= -W - AY + BX. \end{aligned}$$

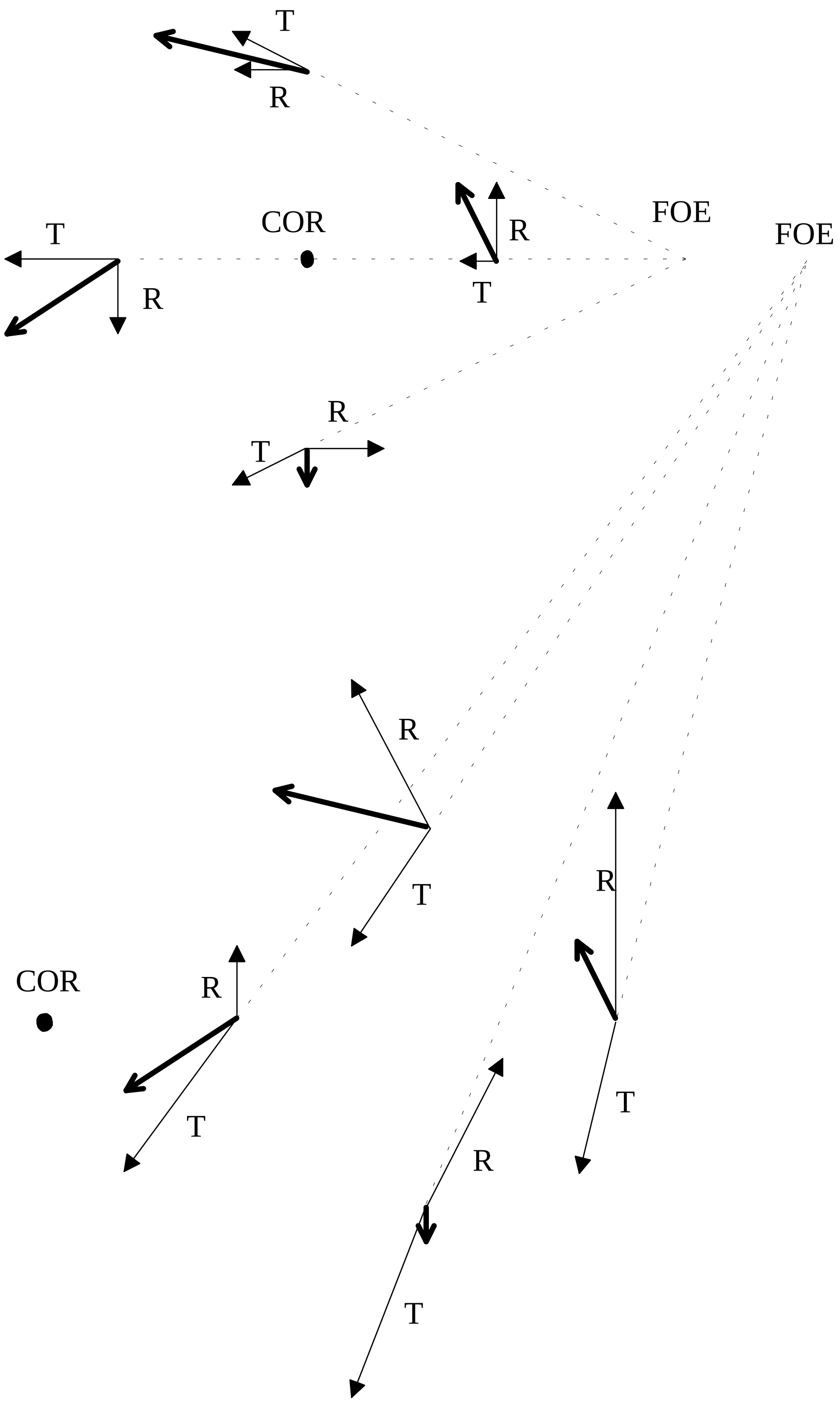

Figure 5. Effect of moving the center-of-rotation (COR) on the focus-of-expansion (FOE) of the same set of optic flow vectors. The dark vectors represent the optic flows. R = rotational component, T = translational component.

We can then express the optical flow *(u,v)* as

$$u = \frac{dx}{dt} = \frac{d}{dt}\left(\frac{Xf}{Z}\right) = \frac{\dot{X}f}{Z} - \frac{X\dot{Z}f}{Z^2} \qquad\qquad v = \frac{dy}{dt} = \frac{d}{dt}\left(\frac{Yf}{Z}\right) = \frac{\dot{Y}f}{Z} - \frac{Y\dot{Z}f}{Z^2}$$

or

$$u = \frac{-Uf + xW}{Z} + A\frac{xy}{f} - B\left(\frac{x^2}{f} + f\right) + Cy \tag{2}$$

$$v = \frac{-Vf + yW}{Z} + A\left(\frac{y^2}{f} + f\right) - B\frac{xy}{f} - Cx.$$

Substituting these definitions of *u* and *v* into the normal flow equation (1), and abbreviating the unit gradient vector, $(n_x, n_y) = \frac{\left(\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}\right)}{\sqrt{\left(\frac{\partial E}{\partial x}\right)^2 + \left(\frac{\partial E}{\partial y}\right)^2}}$, we have

$$U_n = \left[\frac{-Uf + xW}{Z} + A\frac{xy}{f} - B(\frac{x^2}{f} + f) + Cy\right]n_x + \left[\frac{-Vf + yW}{Z} + A(\frac{y^2}{f} + f) - B\frac{xy}{f} - Cx\right]n_y.$$

The only unknown in this equation is *Z*, the depth dimension of the point of interest. Thus, Z can be computed as

$$Z = \frac{(-Uf + xW)n_x + (-Vf + yW)n_y}{U_n - \left[A\frac{xy}{f} - B(\frac{x^2}{f} + f) + Cy\right]n_x - \left[A(\frac{y^2}{f} + f) - B\frac{xy}{f} - Cx\right]n_y} \tag{3}$$

or, in terms of the partial derivatives,

$$Z = \frac{(Uf - xW)\frac{\partial E}{\partial x} + (Vf - yW)\frac{\partial E}{\partial y}}{\frac{\partial E}{\partial t} + \left[A\frac{xy}{f} - B(\frac{x^2}{f} + f) + Cy\right]\frac{\partial E}{\partial x} + \left[A(\frac{y^2}{f} + f) - B\frac{xy}{f} - Cx\right]\frac{\partial E}{\partial y}}.$$

Note that Z cannot be found where $U_n = \left[A\frac{xy}{f} - B(\frac{x^2}{f} + f) + Cy\right]n_x + \left[A(\frac{y^2}{f} + f) - B\frac{xy}{f} - Cx\right]n_y$ (where the normal flow is due entirely to camera rotation). This is because only the translational component of the optical flow (and hence translational component of the normal flow) is dependent on depth.

Once *Z* is found, the other space coordinates are also known, since $X = \frac{xZ}{f}$ and $Y = \frac{yZ}{f}$. Therefore, given any point *(x,y)* in the image and the camera velocities, the position of the corresponding point in the real world (and hence its range) can be computed directly from the normal flow or the spatial and temporal derivatives. Of course, as with all flow methods, the range can be found only for regions of high texture, or at brightness edges, where the derivatives are non-zero. The depth of homogenous regions must be interpolated from the surrounding edges.

Accuracy Analysis

Accuracy of the Normal Flow

Many authors have criticized the accuracy of gradient-derived optical flows [Albus,1990; Barron et al., 1992]. Some of these criticisms also apply to our gradient-based normal flow derivation technique.

Since, from equation (1), the normal flow has magnitude

$$|U_n| = \frac{\left|\frac{\partial E}{\partial t}\right|}{\sqrt{\left(\frac{\partial E}{\partial x}\right)^2 + \left(\frac{\partial E}{\partial y}\right)^2}},$$

we can see that quantization errors and noise-contributed errors will be minimized where both the spatial brightness gradient and temporal change are greatest. Fermuller and Aloimonos [1991] have also shown that the normal flow most accurately represents the normal component of the physical motion field where the brightness gradients are large. Therefore, in practice we should only compute the normal flow at points where the spatial derivatives exceed a minimum threshold. These factors often result in a sparse range map.

Low-pass filtering is necessary for most flow-determination methods [Barron et al., 1992], and especially so for derivative-based techniques, since smoothness in brightness variations is assumed in the derivation of the *optical flow equation*. Low-pass filtering helps enable derivatives to be taken at step edges and helps attenuate the effects of noise and quantization errors, but cannot undo the effects of aliasing due to spatial or temporal frequency components that are higher than the sampling rate. On the other hand, smoothing also removes sharp features that contain the most accurate information.

Albus [1990] noted that besides aliasing problems, smoothing requirements, and sparse output maps, gradient-based techniques also suffer from the non-uniform sensitivity of photodetectors in any array, and from low-frequency thermal drift in detector noise. Even so, derivative-based methods are the simplest and fastest of all flow-determination techniques, and most appropriate for real-time implementation on conventional hardware. The combination of speed and lack of accuracy favors their use in real-time qualitative vision techniques [Aloimonos, 1990].

Computing Derivatives and Gradients in Discrete Domain

There are several ways to compute derivatives in the discrete domain [Rosenfeld & Kak, 1982]. The usual method is to take first-order differences along the desired direction, either using two adjacent pixels (x and x+1) or across the current pixel (x-1 and x+1). The first method produces a poorer approximation [Cheney & Kincaid, 1980], unless the result is associated with the crack between the two pixels. This concept of associating derivatives with cracks between pixels was used by Horn [1986]. He performed the differentiation using first-order differences at the center of a three-dimensional cube. The derivative along any axis is taken as the difference between two slices of image data averaged in a plane perpendicular to that axis (see figure 6).

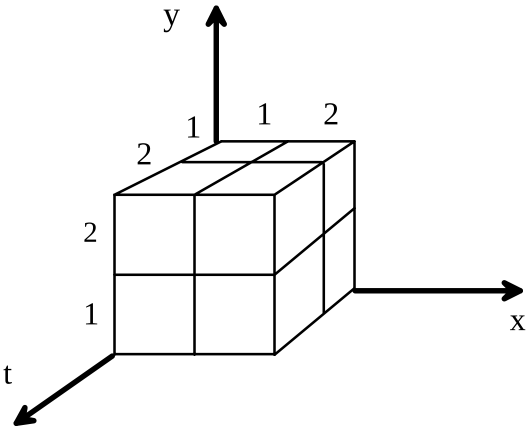

Figure 6. "Slice averaging" method: the derivatives are associated with the center of the cube. A derivative along any one axis is the difference between two slices in the plane perpendicular to that axis. Thus,

$$\frac{\partial E(x,y,t)}{\partial t} \approx \frac{1}{4}\Big[\Big(E_{1,1,2}+E_{1,2,2}+E_{2,1,2}+E_{2,2,2}\Big)-\Big(E_{1,1,1}+E_{1,2,1}+E_{2,1,1}+E_{2,2,1}\Big)\Big]$$

The derivative can be further improved by using 4 or more points (or averaged slices) around the point of interest, instead of the simple difference. For example, a more accurate estimate of the first derivative is [Cheney & Kincaid, 1980]:

$$f'(x_3) \approx \frac{f(x_4)-f(x_2)}{2} - \frac{f(x_5)-2f(x_4)+2f(x_2)-f(x_1)}{12}. \qquad (4)$$

We use the combination of this formula and slice averaging for computing derivatives.

Once the spatial derivatives, $\frac{\partial E}{\partial x}$ and $\frac{\partial E}{\partial y}$, are found, the brightness gradient is normally computed as $\sqrt{\left(\frac{\partial E}{\partial x}\right)^2+\left(\frac{\partial E}{\partial y}\right)^2}$. However, with the square pixel tesselation used by most imaging systems, this often leads to biases in one direction over the others. The biases associated with the two derivative methods (associated with pixels and with cracks between pixels) are illustrated in figure 7. The example shows a step edge at which the gradient should be 1 (and it would be if the edge was vertical or horizontal). However, as shown in the table in figure 7, the gradients associated with this diagonal edge are either over or under 1, caused by errors in the derivatives themselves. These errors are associated with the square pixel tesselation, and can only be eliminated if the camera and framegrabber manufacturers move to another type of tesselation, such as a hexagonal one. This is unlikely to happen in the near future.

	1	1	1	1
y+1	0	1	1	1
y	0	0	1	1
	0	0	0	1
		x	x+1	

	$\partial E/\partial x$	$\partial E/\partial y$	Gradient
Centered on pixel (x,y)	1	1	$\sqrt{2}$
Centered in the middle of the square defined by x, x+1, y, & y+1 (use 2-pixel averaged slices)	1/2	1/2	$1/\sqrt{2}$
Correct values (in continuous domain)	$1/\sqrt{2}$	$1/\sqrt{2}$	1

Figure 7. A diagonal two-dimensional edge showing errors associated with the discrete square pixels.

Errors in Translational Velocities

Assume that there is no rotational motion. Let *(U', V', W')* be the true translational velocities, where

$$(U', V', W') = (U + \Delta U, V + \Delta V, W + \Delta W)$$

and *(U,V,W)* are the velocities used for our computations. Then from Eq. (3), the true Z coordinate should be

$$Z' = \frac{[-(U + \Delta U)f + x(W + \Delta W)]n_x + [-(V + \Delta V)f + y(W + \Delta W)]n_y}{U_n}$$

$$= Z + \frac{(-\Delta U f + x\Delta W)n_x + (-\Delta V f + y\Delta W)n_y}{U_n}$$

$$= Z + \frac{(\Delta U f - x\Delta W)\frac{\partial E}{\partial x} + (\Delta V f - y\Delta W)\frac{\partial E}{\partial y}}{\frac{\partial E}{\partial t}}$$

Thus, for the case of purely translational motion, inaccuracies in the knowledge of *U* and *V* (motion parallel to the image plane) will result in a constant shift (more prominent for longer focal lengths) of the computed depth over the whole image. Inaccuracies in *W* will result in linearly increasing depth errors away from the optical axis.

Errors in Rotational Velocities

We will examine the case where rotation is being kept as close to zero as possible, but not perfectly. Again, from equation (3), the unwanted rotational velocities, $(\Delta A, \Delta B, \Delta C)$, show up in the true range as

$$Z' = \frac{(-Uf + xW)n_x + (-Vf + yW)n_y}{U_n - \left[\Delta A\frac{xy}{f} - \Delta B(\frac{x^2}{f} + f) + \Delta Cy\right]n_x - \left[\Delta A(\frac{y^2}{f} + f) - \Delta B\frac{xy}{f} - \Delta Cx\right]n_y}$$

$$= \frac{(-Uf + xW)n_x + (-Vf + yW)n_y}{U_n}$$

$$\times \frac{1}{1 - \frac{1}{U_n}\left\{\left[\Delta A\frac{xy}{f} - \Delta B(\frac{x^2}{f} + f) + \Delta Cy\right]n_x - \left[\Delta A(\frac{y^2}{f} + f) - \Delta B\frac{xy}{f} - \Delta Cx\right]n_y\right\}}$$

$$= Z \times \frac{1}{1 + \frac{1}{\frac{\partial E}{\partial t}}\left\{\left[\Delta A\frac{xy}{f} - \Delta B(\frac{x^2}{f} + f) + \Delta Cy\right]\frac{\partial E}{\partial x} - \left[\Delta A(\frac{y^2}{f} + f) - \Delta B\frac{xy}{f} - \Delta Cx\right]\frac{\partial E}{\partial y}\right\}}.$$

The right-hand side of the product is the error multiplier, which approaches 1 where U_n (or the temporal change) is large. Thus, we should only look at these pixels for the most accurate results when rotational motion cannot be held to exactly zero.

DERIVING RELATIVE RANGE WITH KNOWN TRANSLATIONAL DIRECTION

In some instances, the exact translational velocities are difficult to obtain, while the direction of travel and rotational motion are much easier to establish. For example, the camera is mounted on a moving platform on a straight rail. The translational direction is the angle between the rail and the camera. The rotational motion is controlled by the camera's pan-and-tilt unit, but the translational velocity is tied to the rail platform and is not easily accessible. In many other instances, while the velocities are difficult to obtain, the straight course of travel of a platform can also be easily accomplished with simple accelerometers, gyroscopes, steering lock, or by applying equal torque to the wheels of a land robot. For these cases, we can still obtain the relative range to various points in the scene.

When there is forward motion (i.e., $W \neq 0$), knowing the direction of travel means that the FOE, the point (x_0, y_0) in the image where the line of travel intersects the image plane, is known.

Since $(x_0, y_0) = (Uf/W, Vf/W)$, we can rearrange equation (3) and get

$$\frac{Z}{W} = \frac{(x - x_0)n_x + (y - y_0)n_y}{U_n - \left[A\frac{xy}{f} - B(\frac{x^2}{f} + f) + Cy\right]n_x - \left[A(\frac{y^2}{f} + f) - B\frac{xy}{f} - Cx\right]n_y}$$

$$= \frac{(x_0 - x)\frac{\partial E}{\partial x} + (y_0 - y)\frac{\partial E}{\partial y}}{\frac{\partial E}{\partial t} + \left[A\frac{xy}{f} - B(\frac{x^2}{f} + f) + Cy\right]\frac{\partial E}{\partial x} + \left[A(\frac{y^2}{f} + f) - B\frac{xy}{f} - Cx\right]\frac{\partial E}{\partial y}}. \qquad (5)$$

Thus, the relative range to points in the image can be obtained with just the normal flow or local derivatives. The exceptions are points where the translational component of the normal flow is zero--and at the FOE, (x_0, y_0) , but the flow is also zero there. The Z/W ratio is known as the *time to adjacency* (the time it takes for an object to impact an infinitely large image plane). In the neighborhood of the FOE (which is where the camera is headed), this ratio is also known as the *time to collision.*

With a relative-range map, the true range to all available points can be computed if the range to one of the points is found. This can be done by many different methods, including simple triangulation using a laser and the same video camera [Nguyen, 1995].

In the absence of rotation, we obtain an even simpler set of equations for the relative range with only direction of motion known:

$$\frac{Z}{W} = \frac{(x - x_0)n_x + (y - y_0)n_y}{U_n} = \frac{(x_0 - x)\frac{\partial E}{\partial x} + (y_0 - y)\frac{\partial E}{\partial y}}{\frac{\partial E}{\partial t}} \qquad (6)$$

for $W \neq 0$.

If the motion is frontal parallel ($W = 0$), then

$$\frac{Z}{Uf} = \frac{-n_x}{U_n} = \frac{\frac{\partial E}{\partial x}}{\frac{\partial E}{\partial t}}$$ for motion along the horizontal axis,

$$\frac{Z}{Vf} = \frac{-n_y}{U_n} = \frac{\frac{\partial E}{\partial y}}{\frac{\partial E}{\partial t}}$$ for motion along the vertical axis, and

$$\frac{Z}{Vf} = \frac{-\frac{U}{V}n_x - n_y}{U_n} = \frac{\frac{U}{V}\frac{\partial E}{\partial x} + \frac{\partial E}{\partial y}}{\frac{\partial E}{\partial t}}$$ for diagonal motion.

TESTING

To test the method, we used the familiar NASA Coke can image sequence (figure 8a) , available from many image archives. The only external information used was that the sequence contains only translational motion, and the motion was toward the center of the Coke can. The images were passed through a Gaussian smoothing filter with convolution stencil:

$$1/110 \bullet \begin{array}{|c|c|c|c|c|} \hline 1 & 3 & 4 & 3 & 1 \\ \hline 3 & 6 & 8 & 6 & 3 \\ \hline 4 & 8 & 10 & 8 & 4 \\ \hline 3 & 6 & 8 & 6 & 3 \\ \hline 1 & 3 & 4 & 3 & 1 \\ \hline \end{array}$$

Three frames were used. One half of the pixel brightness difference between the third and the first frame was used as the temporal derivatives. Since the images were prefiltered, this approximates the "slice difference" method of computing derivatives, but with the results associated with the pixels and not the "cracks." Equation (4) was used on the second frame for computing spatial derivatives. The derivatives were then substituted into equation (6) to compute the relative range. As expected, the method was sensitive to sampling errors. About 5% of the pixels gave negative values for the range. These were obviously erroneous and were discarded. The resulting range image is shown in figure 8b, where dark pixels correspond to farther points, and lighter pixels to progressively closer points. White areas denote locations where no information was available (no temporal or spatial change) or where negative results were obtained.

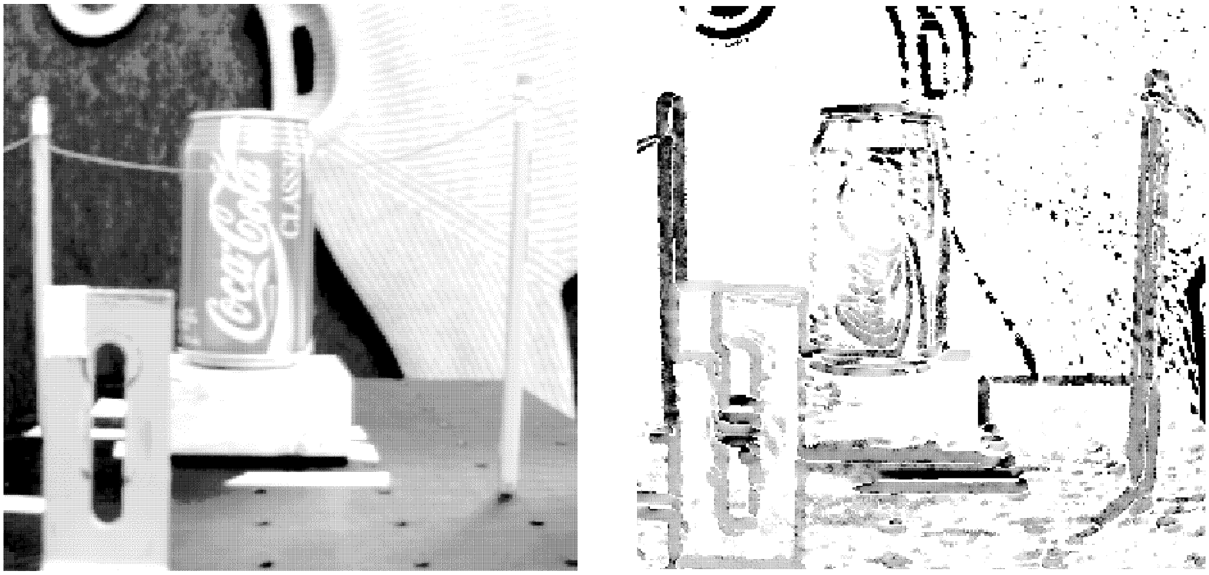

Figure 8. (a) One frame from the NASA Coke can sequence. (b) Range image obtained.

Examining figure 8b, we found that, in general, the results gave correct relative distances to the various objects. The outline of the metal flange was lightest, followed by the pencils and Coke can. The outlines of the sweater and the ring on the back board were darkest. Errors can also be noted. The horizontal dark bar in the range image (under the box) was erroneous and probably due to a combination of the extreme contrast of the white strip in the foreground and the filtering operation (which spreads out a few bad points along the back edge of the strip). However, most of the errors appear to be "salt and pepper" types, and should be easily removed using traditional image processing and computer vision techniques, such as median filtering, region growing, etc.

SUMMARY

Given no knowledge of the motion, deriving range from image motion is a difficult problem. However, in many instances the motion is either known within some degree of accuracy, or the direction of movement is known. We described how range can be computed from a sequence of images given knowledge of the motion, and supplied an analysis of the accuracy of the results based on the accuracy of the known motion. We also discussed how relative range can be computed when only the direction of movement is known, and described an experiment conducted on a sequence of calibrated data.

REFERENCES

Albus, James S. and Tsai Hong Hong, "Motion, Depth, and Image Flow," *Proc. 1990 IEEE Int'l Conf. on Robotics and Automation,* Cincinnati, OH, May 1990.

Aloimonos, J. (Y.), "Purposive and Qualitative Active Vision," *DARPA Image Understanding Workshop,* Pittsburgh, PA, Sept. 1990.

Aloimonos, Yiannis, *Active Perception.* Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers; 1993.

Ballard, Dana H. and Christopher M. Brown, *Computer Vision.* Englewood Cliffs, NJ: Prentice-Hall, 1982.

Barron, J. L et al., "Performance of Optical Flow Techniques," *Proc. 1992 IEEE Computer Vision & Pattern Recognition,* Champaign, IL, June 1992.

Cheney, Ward and David Kincaid, *Numerical Mathematics and Computing.* Monterey, CA: Brooks/Cole Pub. Co, 1980.

Fermuller, Cornelia and Yiannis Aloimonos, "Estimating 3-D Motion from Image Gradients," Technical Report CAR-TR-564. College Park, MD: University of Maryland, Center for Automation Research, June 1991.

Horn, B. K. P., *Robot Vision.* Cambridge, MA: The MIT Press, 1986.

Huang, Liuqing and Yiannis Aloimonos, "Relative Depth from Motion Using Normal Flow: An Active and Purposive Solution," Technical Report CAR-TR-535. College Park, MD: University of Maryland, Center for Automation Research, February 1991.

Nguyen, Hoa G., "A Simple Method for Range Finding via Laser Triangulation," Technical Document 2734. San Diego, CA: Naval Command, Control and Ocean Surveillance Center, RDT&E Division, January 1995.

Rosenfeld, Azriel and Avinash C. Kak, *Digital Picture Processing*, Volume 2. New York, NY: Academic Press, 1982.