A Comparison of 2 Methods for Recovering Dense Accurate Depth Using Known 3D Camera Motion

Baozhong Tian John Barron Wang Kay Jacky Ngai
Dept. of Computer Science
Univ. of Western Ontario
London, Ontario, Canada, N6A 5B7
btian,barron@csd.uwo.ca

Hagen Spies
Computer Vision Lab, Linköping University,
583 31 Linköping, Sweden
hspies@isy.liu.se

Abstract We present a comparison of two Kalman filter frameworks by Ngai, Barron and Spies [1] and Hung and Ho [6] for recovering depth from the time-varying optical flow fields generated by a camera translating over a scene by a known amount. Synthetic data made from ray traced cubical, cylindrical and spherical primitives are used in the optical flow calculation and a quantitative error analysis of the recovered depth shows our approach is significantly better.

Keywords: Depth Map from Time-Varying Optical Flow/Intensity Derivatives, Kalman Filter, Known Camera Motion, Quantitative Error Analysis

1 Introduction

We consider the problem of depth recovery from monocular image sequences when the 3D camera motion is either known [6] or recovered from a Motion and Structure algorithm [3]. Using the assumption of local planarity, depth can first be computed from the measured optical flow [1] or from image intensity derivatives [6] in a Kalman filter framework

There are a number of approaches for computing depth from optical flow and/or image derivatives in the literature using a Kalman filter, see [1]. Recent work has used Kalman filtering as a way to recover depth values from time varying flow where the camera motion is known [6, 1]. Kalman filtering seems especially appropriate here as both optical flow and image differentiation errors seems Gaussian [1].

2 Literature Survey

A complete survey of dense depth recovery methods can be found in Ngai et al. [1] and a M.Sc. thesis [2]. We chose to compare our algorithm to Hung and Ho's, because their algorithm is very recent (1999), they seem to obtain very good qualitative results and we can make good synthetic data approximating their real data (our cylinder and cube data).

3 Ngai, Barron and Spies

We also assume that the direction of sensor translation, $\hat{u}$, and its rotation, $\vec{\omega}$, are known or can be computed by some other means [3]. In the case where the true sensor translation $\vec{U}$ is known, we can compute the absolute 3D depth map; otherwise we can compute relative depth. Interpolation of surface orientation values at non-pixel image locations is avoided by assuming local planarity, a seemingly reasonable assumption everywhere in the image except at depth discontinuities. We use planarity to avoid having to compute non-pixel correspondences. Given image velocity $\vec{v}(\vec{Y}, t)$ we believe it is valid most of the time to assume the correspondence is at round$(\vec{Y} + \vec{v}\delta t)$ at time $t + \delta t$.

We consider a setup consisting of a single camera taking the images while it is moving through a static 3D scene. The standard image velocity equations [7] relate a velocity vector measured at image location $\vec{Y} = (y_1, y_2, f) = f\vec{P}/X_3$, [i.e. the perspective projection of a 3D point $\vec{P} = (X_1, X_2, X_3)$], to the 3D sensor translation $\vec{U}$ and 3D sensor rotation $\vec{\omega}$. We can rewrite the standard equation for the image velocity $\vec{v}=(v_1,v_2)$ as $\vec{v}(\vec{Y},t)=\vec{v}_T(\vec{Y},t)+\vec{v}_R(\vec{Y},t)$ where $\vec{v}_T$ and $\vec{v}_R$ are the translational and rotational components of image velocity:

$$\vec{v}_T(\vec{Y},t) = A_1(\vec{Y})\frac{\vec{U}}{X_3} \quad \text{and} \quad \vec{v}_R(\vec{Y},t) = A_2(\vec{Y})\vec{\omega}(t), \tag{1}$$

with

$$A_1(\vec{Y}) = \begin{pmatrix} -f & 0 & y_1 \\ 0 & -f & y_2 \end{pmatrix} \text{ and} \tag{2}$$

$$A_2(\vec{Y}) = \begin{pmatrix} \frac{y_1 y_2}{f} & -(f+\frac{y_1^2}{f}) & y_2 \\ (f+\frac{y_2^2}{f}) & -\frac{y_1 y_2}{f} & -y_1 \end{pmatrix}. \tag{3}$$

We define the depth scaled camera translation as

$$\vec{u}(\vec{Y},t) = \frac{\vec{U}(t)}{||\vec{P}(t)||_2} = \hat{u}\mu(\vec{Y},t), \tag{4}$$

where $\hat{u}=\hat{U}=(u_1,u_2,u_3)$ is the normalized direction of translation and $\mu(\vec{Y},t)=\frac{||\vec{U}||_2}{||\vec{P}||_2}=\frac{f||\vec{U}||_2}{|X_3|\,||\vec{Y}||_2}$ is the depth scaled sensor speed at $\vec{Y}$ at time t. We refer to μ as **scaled** speed (or relative depth). Note that translational image velocity, $\vec{v}_T$, is bilinear in $\hat{u}$ and μ, making the image velocity equations nonlinear. The focal length f is assumed to be known via some camera calibration scheme. If we define 2 vectors:

$$\vec{r}(\vec{Y}) = (r_1,r_2) = |\vec{v}-A_2(\vec{Y})\vec{\omega}| \quad \text{and} \tag{5}$$

$$\vec{d}(\vec{Y}) = (d_1,d_2) = |A_1(\vec{Y})\hat{u}|\frac{||\vec{Y}||_2}{f}, \tag{6}$$

where $|\vec{A}|$ means each element in the vector is replaced by its absolute value. Then we can solve for μ from the image velocity equation, which can now be written as $\vec{r}-\vec{d}\mu=(r_1,r_2)-(d_1,d_2)\mu=0$. μ has solutions $\frac{r_1}{d_1}$ or $\frac{r_2}{d_2}$ or (best) a weighted average:

$$\mu = \frac{\left(\frac{r_1|v_1|}{d_1}+\frac{r_2|v_2|}{d_2}\right)}{|v_1|+|v_2|}. \tag{7}$$

We use the magnitudes of v_1 and v_2 to weight the calculation: we expect μ values computed from the larger velocity component magnitude to be more reliable.

3.1 Planar Orientation from Relative Depth

We are interested in computing the local surface orientation as a unit normal vector, $\hat{\alpha}=(\alpha_1,\alpha_2,\alpha_3)$ from μ values. Consider two 3D points, $\vec{P}_1=(X_{11},X_{12},X_{13})$ and $\vec{P}_2=(X_{21},X_{22},X_{23})$, with images $\vec{Y}_1=\left(\frac{fX_{11}}{X_{13}},\frac{fX_{12}}{X_{13}},f\right)$ and $\vec{Y}_2=\left(\frac{fX_{21}}{X_{23}},\frac{fX_{22}}{X_{23}},f\right)$. If they lie on the same 3D plane then:

$$\frac{\hat{\alpha}\cdot\vec{Y}_1}{\hat{\alpha}\cdot\vec{Y}_2}=\frac{X_{23}}{X_{13}}. \tag{8}$$

This equation gives the ratio of the 3^{rd} coordinates (X_3) of two 3D points in terms of their image locations and their planar surface orientation (assuming they lie on a common 3D plane). From the definition of $\mu=\frac{||\vec{U}||_2}{||\vec{P}||_2}=\frac{f||\vec{U}||_2}{|X_3|\,||\vec{Y}||_2}$ we can write:

$$X_3=\frac{f||\vec{U}||_2}{\mu||\vec{Y}||_2}. \tag{9}$$

From the planar equation $\hat{\alpha}\cdot\vec{P}=\hat{\alpha}\cdot\frac{X_3}{f}\vec{Y}=c$ and using equations (9) and (8) we obtain:

$$\hat{\alpha}\cdot\vec{Y}=\frac{c\mu||\vec{Y}||_2}{||\vec{U}||_2} \tag{10}$$

We can solve for $\frac{\hat{\alpha}}{c}$ by setting up a linear system of equations, one for each pixel in a $n\times n$ neighbourhood where planarity has been assumed and using a standard least squares solution method.

3.2 The Overall Calculation

If we assume that $\hat{u}$ (or indeed $\vec{U}$) and $\vec{\omega}$ are knowns, we need only concern ourselves with the surface orientation step of the calculation.

At the initial time, $t=1$:

1. Given $\hat{u}$ and $\vec{\omega}$, we compute all the μ's as described above (see [3] for one way of computing $\hat{u}$ and $\vec{\omega}$ from time-varying optical flow).

2. In each $n\times n$ neighbourhood centered at a pixel (i,j) we compute $(\frac{\hat{\alpha}}{c})_{(i,j)}$ at that pixel using equations (7) and (10). We call these computed $\frac{\hat{\alpha}}{c}$'s the measurements and denote them as $\vec{g}_{M_{(i,j)}}$.

3. Given these measurements, we use the $\vec{g}_{M_{(i,j)}}$ to recompute the $\mu_{(i,j)}$'s as:

$$\mu(i,j)=\frac{(\vec{g}_{M_{(i,j)}}\cdot\vec{Y}_{(i,j)})||\vec{U}||_2}{||\vec{Y}_{(i,j)}||_2} \tag{11}$$

The recomputed μ values are more "smoothed" than the actual measurements and better represent the scene shape. These $\mu(i,j)$ values are currently the best estimate of the scene's dense shape. Note that we can obtain μ values for pixels with no optical flow, these are computed from the image velocities in its neighbourhood (which are assumed to result from the same local planar patch). If smoothing is turned on, we apply a median filter to the $\mu(i,j)$ within 5×5 neighbourhoods. This will remove outliers. We repeat step 2 except that now instead of using the measured the $\mu(i,j)$, we use the already smoothed $\mu(i,j)$ in equation (10). Now the $\vec{g}_{M_{(i,j)}}$'s are recomputed, yielding a significantly smoother surface.

At time $t \geq 2$:

1. Given the measurements or best estimates of $\hat{u}$ and $\vec{\omega}$, we compute μ at each pixel location and then compute all $\vec{g}_{M_{(i,j)}}$'s in the same way described above for the new optical flow field. Using the image velocity measurements at time $t = i$, we use the best estimate of surface orientation at time $t = i - 1$ at location $\vec{Y} - \vec{v}$ ($\Delta t = 1$) plus the measurement at $\vec{Y}$ and its covariance matrix to obtain a new best estimate at $\vec{Y}$ at time $t = i$. We do this at all $\vec{Y}$ locations (where possible), recompute the μ values via equation (11) and output these as the 3D shape of the scene. Of course, if smoothing is enabled we also perform that operation.

At time $t = i$ we proceed as for time $t = 2$, except we use the best μ estimates from time $t = i - 1$ instead of time $t = 1$ in the Kalman filter updating.

Note that if the true sensor translation $\vec{U}$ is known, the absolute 3D depth X_3 can be computed everywhere from the filtered μ values:

$$X_3 = \frac{f||\vec{U}||_2}{\mu||\vec{Y}||_2}. \tag{12}$$

3.3 The Kalman Filter Equations

We note here that the components of $\frac{\hat{\alpha}}{c}$ in equation (10) are not independent, thus we have a covariance matrix with non-zero off diagonal elements in the Kalman filter equations. [In [3], the components of all 2D and 3D vectors, $\hat{u}$ and $\vec{\omega}$ respectively were treated as 1D variables in the Kalman filter framework.]

If we assume $\hat{u}$ and $\vec{\omega}$ are known, we can compute $\frac{\hat{\alpha}}{c}$ at each pixel (i,j) as outlined above, assuming local planarity. For the purposes of understanding the equations below we will subscript symbols with a M to indicate measured quantities (computed from the image velocities), P to indicate predicted quantities and C to indicate the computed quantities (the current best estimates). We use $\vec{g}$ to denote $\frac{\hat{\alpha}}{c}$. It is a 3D quantity, so we need an initial predicted value and an initial covariance matrix at time $t = 0$:

$$\vec{g}_{P_{(i,j)}} = \vec{0}, \tag{13}$$

$$C_{P_{(i,j)}} = \begin{bmatrix} \infty & 0 & 0 \\ 0 & \infty & 0 \\ 0 & 0 & \infty \end{bmatrix}. \tag{14}$$

This definition of C_P says that initially the coefficients of $\vec{g}$ are independent and we have no confidence in their estimates.

For each time $t = 1, 2, 3, ...$ at all pixels (i,j), we use equations (7) and (10) to make a measurement of $\vec{g}_{M_{(i,j)}}$ and an estimate of its covariance matrix $C_{M_{(i,j)}}$, respectively. Then using the previous best surface orientation estimate at time $t - \Delta t$ at image location $\vec{Y} - \vec{v}\Delta t$, denoted by (i^-, j^-), the Kalman filter equations are computed as follows:

$$K_{(i,j)} = C_{P_{(i^-,j^-)}} \left[C_{P_{(i^-,j^-)}} + C_{M_{(i,j)}} \right]^{-1}, \tag{15}$$

$$\vec{g}_{C_{(i,j)}} = \vec{g}_{P_{(i^-,j^-)}} + K_{(i,j)} \left(\vec{g}_{M_{(i,j)}} - \vec{g}_{P_{(i^-,j^-)}} \right) \tag{16}$$

$$C_{C_{(i,j)}} = C_{P_{(i^-,j^-)}} - K_{(i,j)} C_{P_{(i^-,j^-)}} \tag{17}$$

Because $\vec{g}$, i.e. $\frac{\hat{\alpha}}{c}$, can be rotated by sensor rotation, the predicted values $\vec{g}_{P_{(i,j)}}$ must take this into account in their update:

$$\vec{g}_{P_{(i,j)}} = R(\vec{\omega}, t + \Delta t) R^T(\vec{\omega}, t) \vec{g}_{C_{(i,j)}}, \tag{18}$$

$$C_{P_{(i,j)}} = C_{C_{(i,j)}}. \tag{19}$$

3.4 Abnormal Situations and Their Resolution

While tracking individual surface orientations, a number of situations may arise. It is possible that:

1. When tracking a surface orientation at a moving pixel, no surface orientation can be computed at the latest time, in which case tracking stops.

2. A new, untracked surface orientation can be computed, in which case tracking starts.

3. A surface orientation is tracked to a wrong pixel, in which case the tracking continues from that wrong pixel as no error recovery is possible or detectable.

4. The surface orientations at two different pixels with (perhaps) different surface orientations track to the same pixel, in which case the surface orientations are combined and then tracked as a single surface orientation (a weighted average using the covariance matrices as weighting matrices).

4 Hung and Ho

Hung and Ho's approach [6] is a recent dense depth calculation from image intensity derivatives. Hung and Ho use a direct approach which does not require the estimation of optical flow as an intermediate step to recover a dense depth map from a sequence of monocular images with known camera motion. They assume a focal length, $f = 1$, for the camera model with perspective projection (depth measurements are in f units). In their approach, they first let $I(x, y, t)$ be the intensity of the image point (x, y) at time step t in the image sequence and let I_x, I_y, I_t be the partial derivatives of $I(x, y, t)$ with respect to x, y and t, respectively. They assume that the image intensity of corresponding points in the 3D scene is not changed by motion over the image sequence, then expansion of the total derivative of the image intensity $I(x, y, t)$ leads to the motion constant equation, $I_x u + I_y v + I_t = 0$. Hung and Ho assume a known camera translational velocity of $\vec{U} = (U_1, U_2, U_3)$ and a rotational velocity of $\vec{\omega} = (\omega_1, \omega_2, \omega_3)$. The standard image velocity equations show the sensor's motion can be related to the spatio-temporal derivatives of the intensity function as follows:

$$\frac{\vec{s} \cdot \vec{U}}{Z} + \vec{q} \cdot \vec{\omega} = -I_t, \tag{20}$$

where $\vec{s} = (-I_x, -I_y, xI_x + yI_y)$ and $\vec{q} = (xyI_x + (1+y^2)I_y, -xyI_y - (1+x^2)I_x, yI_x - xI_y)$. Since both $\vec{U}$ and $\vec{\omega}$ are known, the depth Z can be estimated directly from the intensity derivatives using equation (20):

$$Z = -\frac{\vec{s} \cdot \vec{U}}{\vec{q} \cdot \vec{\omega} + I_t} \tag{21}$$

Since the camera is moving, the projection of a 3D point on the image plane will change from the image point (x, y) at time t to image point $(x + \Delta x, y + \Delta y)$ at time $(t + \Delta t)$, where Δx and Δy are image point displacements in the amount of time Δt. Let $P(x, y, t) = (X(x, y, t), Y(x, y, t), Z(x, y, t))$ be the corresponding 3D point of the image point (x, y) at time step t. Then

$$Z(x + \Delta x, y + \Delta y, t + \Delta t) = Z(x, y, t) -$$
$$(U_3 \Delta t + \omega_1 \Delta t Y(x, y, t) - \omega_2 \Delta t X(x, y, t)). \tag{22}$$

Hung and Ho assume local smoothness in the depth map and use a Taylor series expansion in the first two arguments of equation (22) to get:

$$Z(x + \Delta x, y + \Delta y, t + \Delta t) = Z(x, y, t + \Delta t) +$$
$$\frac{\partial Z}{\partial x}\Delta x + \frac{\partial Z}{\partial y}\Delta y + e(x, y, t + \Delta t), \tag{23}$$

where $e(x, y, t + \Delta t)$ represents the approximation error. Equating equations (22) and (23) gives:

$$Z(x, y, t + \Delta t) = (1 - \omega_1 \Delta t y + \omega_2 \Delta t x) Z(x, y, t) -$$
$$U_3 \Delta t - \frac{\partial Z}{\partial x}\Delta x - \frac{\partial Z}{\partial y}\Delta y - e(x, y, t + \Delta t), \tag{24}$$

where the perspective projection equations $X(x, y, t) = Z(x, y, t)x$ and $Y(x, y, t) = Z(x, y, t)y$ are used under the assumption $f = 1$. By denoting $Z(t)$ to be $Z(x, y, t)$ for a fixed image point (x, y), and replacing t and $t + \Delta t$ by k and $(k + 1)$ for the k^{th} and $(k+1)^{th}$ image sampling instant, equation (24) can then be written as:

$$Z(k + 1) = G(k)Z(k) + u(k) + \theta(k), \tag{25}$$

where $G(k) = 1 - \omega_1 \Delta t y + \omega_2 \Delta t x$, $u(k) = -U_3 \Delta t - \frac{\partial Z}{\partial x}\Delta x - \frac{\partial Z}{\partial y}\Delta y$ and $\theta(k)$ is taken to include $-e(x, y, k + 1)$ as well as the error generated when estimating the terms $\frac{\partial Z}{\partial x}\Delta x$ and $\frac{\partial Z}{\partial y}\Delta y$ in $u(k)$. The terms $\frac{\partial Z}{\partial x}$ and $\frac{\partial Z}{\partial y}$ can only be estimated after the depth map has attained some degree of smoothness. θ is approximately a Gaussian random noise with zero mean and variance Q. By introducing a measurement noise n_1 with variance R_1 [1] into equation (20), equation (20) can re-written as:

$$Y_1(k) = H_1(k)Z(k) + n_1(k) \tag{26}$$

where $Y_1(k) = -\vec{s} \cdot \vec{U}$ and $H_1(k) = \vec{q} \cdot \vec{\omega} + I_t$.

4.1 Incorporating Surface Structure

Hung and Ho's approach assumes that the depth $Z(x, y)$ for every particular point (x, y) in the image has some local structural property among its neighbouring pixels, which can be expressed as:

$$Z(x, y) = g(Z(x+p_1, y+q_1), ..., Z(x+p_s, y+q_s)) - E, \tag{27}$$

where g is a function defined over a mask indexed by p_i and q_i, $(i = 1, ..., s)$ around the image point (x, y), and E represents permissable variation in the local surface structure with variance R_E. If the estimated depths, say Z_e, of neighbouring pixels are known, an estimate, say Y_2, for $Z(x, y)$ based on this a priori known structure of the surface can be expressed as:

$$Y_2 = g(Z_e(x+p_1, y+q_1), ..., Z_e(x+p_s, y+q_s)). \tag{28}$$

Let p be the error in estimating $Z(x, y)$ arising from the replacement of Z by Z_e in $g()$ be as follows:

$$p = g(Z_e(x + p_1, y + q_1), ..., Z_e(x + p_s, y + q_s)) -$$

[1] We write R_1, as used by Hung and Ho, rather than the standard notation σ to denote variance.

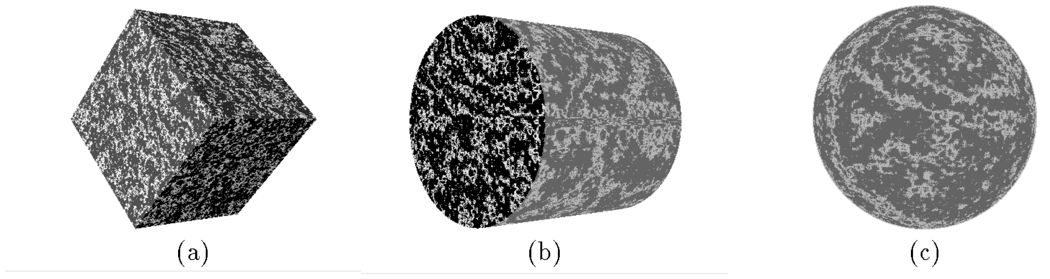

Figure 1: Synthetic test data: (a) A marble-texture cube with sides of length 300 with its center located at (0,0,1500), (b) A marble-texture cylinder with two end spheres of radius 200 and a wall of length 400 with its center located at (0,0,1500) and (c) A marble-texture sphere of radius 200 with its center located at (0,0,1300).

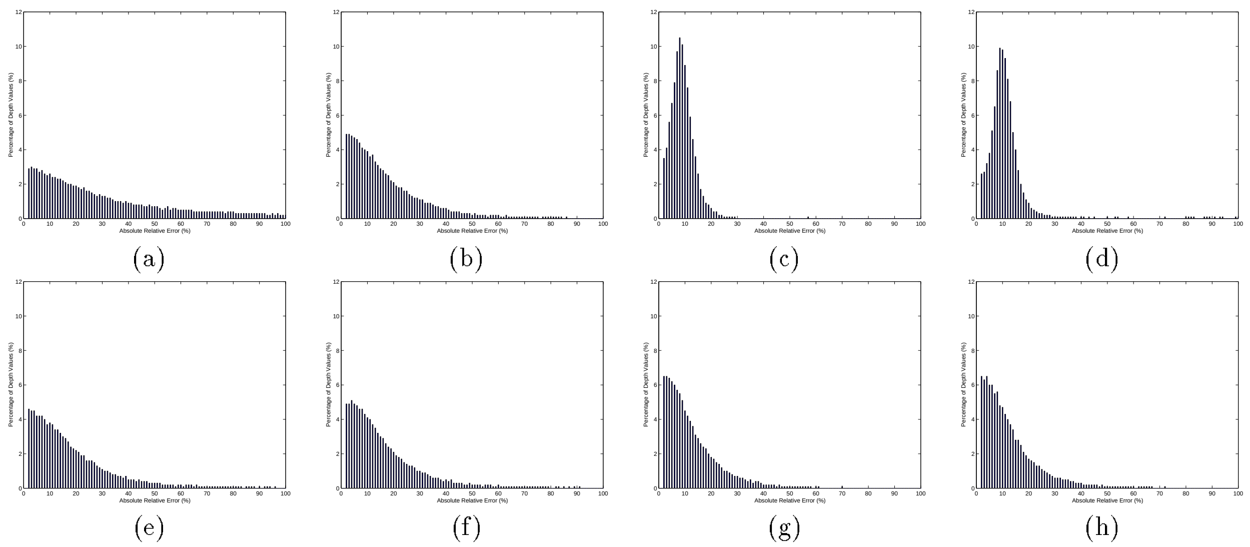

Figure 2: The histogram of the absolute relative errors of the estimated depth values at the 7^{th}, 19^{th}, 27^{th} and 36^{th} images of the **sphere** sequence using Hung and Ho's method [(a) to (d)] and Ngai, Barron and Spies' method [(e) to (h)].

$$g(Z(x+p_1, y+q_1), ..., Z(x+p_s, y+q_s)). \tag{29}$$

Then subtracting equation (27) from equation (28) yields:

$$Y_2 = Z(x,y) + n_2, \tag{30}$$

where $n_2 = E{+}p$. Hung and Ho found during the implementation of their approach that some estimated depths (e.g. $Z_e(x+1,y)$ and $Z_e(x,y+1)$) are unavailable at the time when the depth at pixel (x,y) is being updated. In this case, they compute $Y_2(x,y)$ for a pixel (x,y) as follows:

$$Y_2(x,y) = \frac{1}{2}\left[Z_e(x-1,y) + Z_e(x,y-1)\right]. \tag{31}$$

Hung and Ho say this measurement is spatially biased and may produce effects in propagation of depth estimates due to the diagonalisation of Y_2 values. To overcome this, each image frame is filtered four times starting from a different corner of the image each time and working row by row, producing with four different versions of Y_2, say Y_2^1, Y_2^2, Y_2^3 and Y_2^4. That is, Y_2^1 is generated by filtering (averaging by equation (31)) the image from left to right starting from the top left corner and working down from the top row to the bottom row; Y_2^2 is generated by filtering the image from right to left starting from the top right corner and working down from the top row to the bottom row; Y_2^3 is generated by filtering the im-

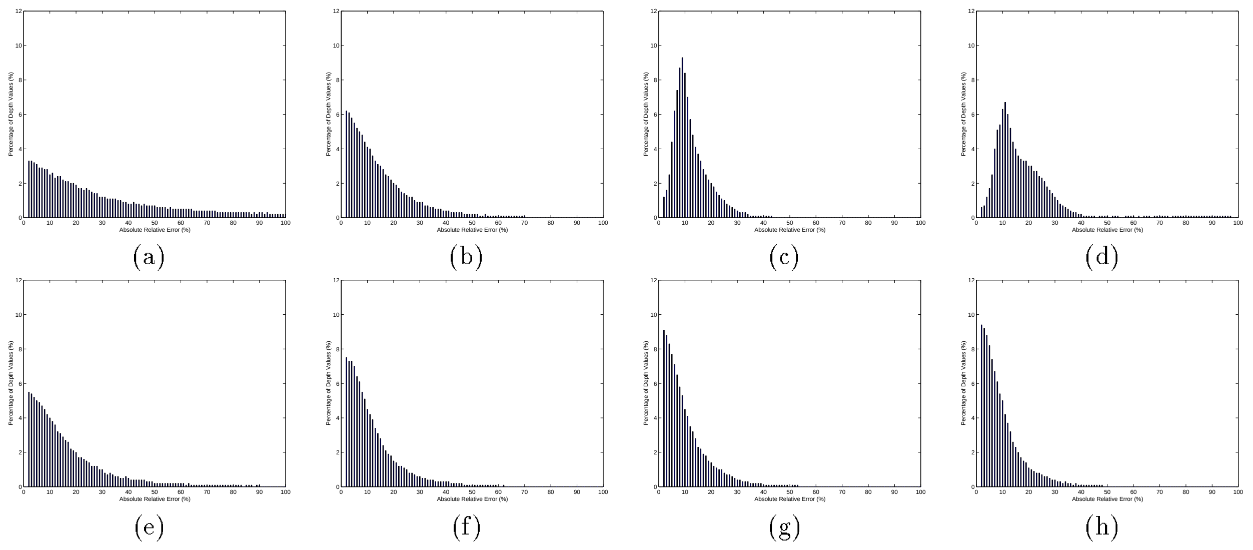

Figure 3: The histogram of the absolute relative errors of the estimated depth values at the 7^{th}, 19^{th}, 27^{th} and 36^{th} images of the **cylinder** sequence using Hung and Ho's method [(a) to (d)] and Ngai, Barron and Spies' method [(e) to (h)].

age from left to right starting from the bottom left corner and working up from the bottom row to the top row; Y_2^4 is generated by filtering the image from right to left starting from the bottom right corner and working up from the bottom row to the top row. Note that this final result is dependent on the filtering order in each case. The final estimate for Y_2 is then taken to be:

$$Y_2 = \frac{1}{4}\left[Y_2^1 + Y_2^2 + Y_2^3 + Y_2^4\right]. \quad (32)$$

By assuming E and p are independent of each other, the variance R_2 of n_2 can be computed as the sum of their variances, $R_2 = R_E + R_p$. R_E is set to an appropriate value by empirical means in each particular experiment because it directly controls the compromise (in the Kalman filter) between the measurement Y_1 from the image intensity derivatives and the measurement Y_2 from the surface structural assumption. R_p can be expressed in terms of the variances of the estimation errors in equation (31). Thus, since from equations (29) and (31),

$$p = \frac{1}{2}\left[(Z_e(x-1,y) - Z(x-1,y)) + (Z_e(x,y-1) - Z(x,y-1))\right], \quad (33)$$

we obtain:

$$R_p = \frac{a}{4}\left[P(x-1,y) + P(x,y-1)\right], \quad (34)$$

where a is a factor to compensate for any under-estimation of R_p due to the surface structural assumption. By empirical means, a suitable range for a is found to be between 1.6 and 2.0 inclusively.

4.2 The Kalman Filter equations

With the extra measurement Y_2, equations (26) and (30) can be combined to give:

$$Y = HZ + n, \quad (35)$$

where

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}, \text{H} = \begin{bmatrix} H_1 \\ 1 \end{bmatrix} \text{ and n } = \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}. \quad (36)$$

The covariance matrix R of n is defined as:

$$R = var(n) = diag(R_1, R_2). \quad (37)$$

Based on equation (35), the set of Kalman filter equations for generating an estimate $Z_e(k)$ for the depth $Z(k)$ in Hung and Ho's approach is then given by as follows:

$$\begin{aligned} Z_e^-(k) &= G(k-1)Z_e(k-1) + u(k-1), && (38)\\ P^-(k) &= G(k-1)P(k-1)G^T(k-1) + Q(k-1), && (39)\\ K(k) &= P^-(k)H^T(k)\left[H(k)P^-(k)H^T(k) + R(k)\right]^{-1} && (40)\\ Z_e(k) &= Z_e^-(k) + K(k)(Y(k) - H(k)Z_e^-(k)) \text{ and} && (41)\\ P(k) &= P^-(k) - K(k)H(k)P^-(k), && (42)\end{aligned}$$

where $Z_e^-(k)$ is the predicted depth at image sampling instant k before the arrival of the k-th measurement, $P^-(k)$ and $P(k)$ are the variances of $Z_e^-(k)$

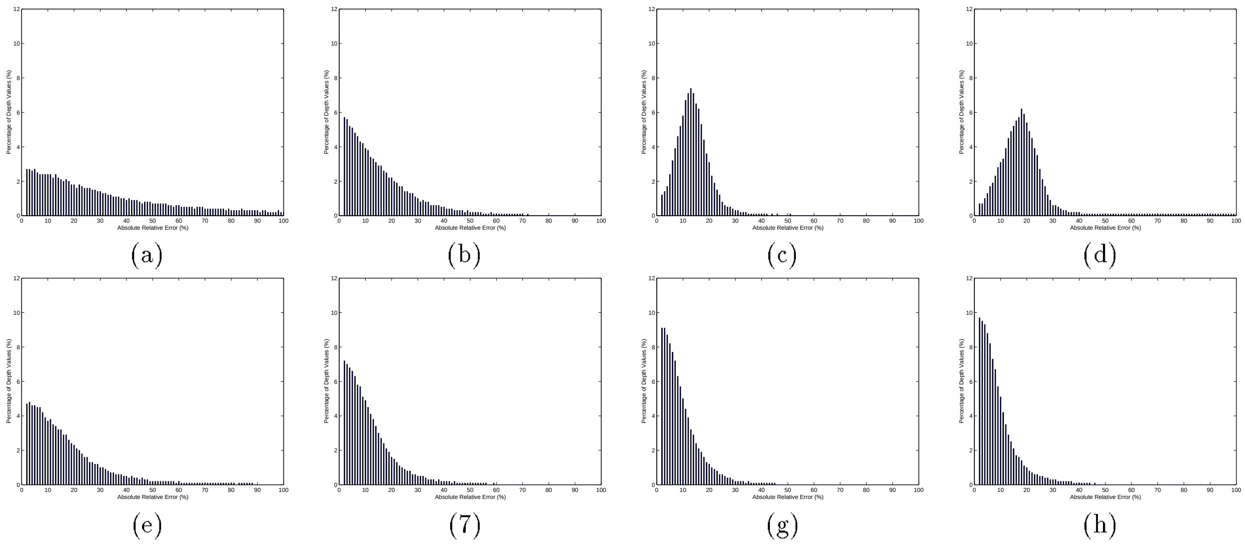

Figure 4: The histogram of the absolute relative errors of the estimated depth values at the 7^{th}, 19^{th}, 27^{th} and 36^{th} images of the **cube** sequence using Hung and Ho's method [(a) to (d)] and Ngai, Barron and Spies' method [(e) to (h)].

and $Z(k)$, respectively, and $K(k)$ is the Kalman filter gain. At $k = 0$, variables can be arbitrarily initialized (we use Hung and Ho's values).

5 Generation of Synthetic Test Image Sequences

For the experiments, we use synthetic images because that allows for a quantitative error analysis on the estimated depth values. We generate a 30 image sequence of 512×512 images using each of three different 3D objects: a sphere, a cube and a cylinder (Figures 1a, 1b and 1c). For each experiment, a sequence of images of one of these three objects is generated while the synthetic camera is moving in a known way and the object is rendered by perspective projection onto the image plane. In all the experiments we performed, all the image sequences were generated using a camera translation of $(1, 1, 0)$ with a focal length of 1000. All the objects have a marble texture to facilitate optical flow computation. Although the synthetic images themselves are error free, the optical flow computed using them is not [1]. All the objects have a marble texture, allowing optical flow to be computed easily because the high variation in local image intensity structure attenuates the aperture problem.

6 Experimental Results and Discussion

Error was measured for relative depth μ using the exact μ values: hence the use of the term absolute relative error. Optical flow was computed using Lucas and Kanade's algorithm [5] with differentiation performed as proposed by Simoncelli [8]. All the figures (2 to 4) show the error histograms for the 7^{th}, 19^{th}, 27^{th} and 36^{th} frames for the 3 objects. For Hung and Ho's method, Gaussian smoothing was incorporated into the Kalman filter after the 19^{th} frame. Figure 2 shows the histograms of the error distribution for the sphere data while Figure 3 shows the histograms of the error distribution for the cylinder data and, finally, Figure 4 shows the histograms of the error distribution for the cube data. Note that at depth discontinuities, where the local planarity assumption is definitely violated, no depth is recovered as the Kalman filter rejects those values as unreliable. This also supposes image velocities can be computed at depth discontinuities. Normally Lucas and Kanade optical flow does not yield velocity values at discontinuities.

Table 1 shows the average absolute relative error for the 3 objects for a number of ranges. The quantitative error results show Ngai, Barron and Spies; results are better in all cases. The results for the sphere (the most curved objects) show the least improvement (but they are still better). A qualitative

Percent Relative Error	>15%	5% - 15%	<5%	>15%	5% - 15%	<5%
Algorithm	Hung and Ho			Ngai, Barron and Spies		
7th depth map (Sphere)	70.72%	18.42%	10.86%	67.51%	19.99%	12.50%
19th depth map (Sphere)	40.96%	35.40%	23.64%	64.04%	21.84%	14.13%
27th depth map (Sphere)	7.33%	65.02%	27.65%	58.02%	23.76%	18.23%
36th depth map (Sphere)	15.18%	68.15%	16.67%	58.03%	23.96%	18.01%
7th depth map (Cylinder)	66.91%	20.33%	12.76%	48.84%	29.96%	21.20%
19th depth map (Cylinder)	33.26%	38.03%	28.71%	37.46%	33.56%	28.98%
27th depth map (Cylinder)	21.70%	62.39%	15.91%	33.45%	32.98%	33.57%
36th depth map (Cylinder)	43.11%	50.21%	6.68%	31.01%	33.74%	35.25%
7th depth map (Cube)	70.69%	18.54%	10.78%	43.21%	34.65%	22.14%
19th depth map (Cube)	37.20%	36.48%	26.32%	26.23%	41.28%	32.48%
27th depth map (Cube)	29.64%	60.44%	9.92%	17.62%	41.34%	41.04%
36th depth map (Cube)	58.30%	36.52%	5.18%	16.10%	40.10%	43.80%

Table 1: The percentage of the estimated depth values that have certain absolute relative errors in the experiments for the sphere, cylinder and cube, using Hung and Ho's and Ngai, Barron and Spies' algorithm, both with smoothing after the 19th frame.

examination of depth values for the 2 methods shows that the magnitudes of the depth values for Hung and Ho's method tend to get smaller overall as the frame number increases (after smoothing has been incorporated): this effect shows up by the error distribution peak moving slightly to the right, as can be seen in the Hung and Ho histograms in Figures 2, 3 and 4. This effect is under active investigation.

7 Conclusions

We have presented a new algorithm to compute dense accurate depth using a Kalman filter framework [1] and showed superior performance to that of Hung and Ho [6]. If we turn on smoothing at the 7^{th} rather than 20^{th} frame, our results become marginally better still. We need to test our algorithm for camera motions including rotation and to use real data (preferably with ground truth). Lastly, we plan to integrate this algorithm into the Kalman filter based motion and structure algorithm designed earlier [3].

References

[1] W.K.J. Ngai, J.L. Barron and H. Spies (2003), Quantitative Depth Recovery from Time-Varying Optical Flow in a Kalman Filter Framework", "Theoretical Foundations of Computer Vision, Geometry, Morphology, and Computational Imaging", LNCS (in press), 2003.

[2] W.K.J. Ngai, "A comparison of 2 methods for recovering dense accurate depth using known 3D camera motion", M.Sc. Thesis, Dept. of Computer Science, The University of Western Ontario, 2002.

[3] Barron J.L. and Eagleson R. (1996), "Recursive Estimation of Time-Varying Motion and Structure Parameters", *Pattern Recognition*, Vol. 29, No. 5, May, pp. 797-818.

[4] Gelb A. (1974), "Applied Optimal Estimation", MIT Press.

[5] Lucas, B. and Kanade, T. (1981), "An iterative image registration technique with an application to stereo vision", *Proc. DARPA IU Workshop*, pp. 121-130.

[6] Hung Y.S. and Ho H.T. (1999), "A Kalman Filter Approach to Direct Depth Estimation Incorporating Surface Structure", *IEEE PAMI*, June, pp. 570-576.

[7] Longuet-Higgins H.-C. and Prazdny K, (1980), "The Interpretation of a Moving Retinal Image", Proc. R. Soc. Lond B 208, pp. 385-397.

[8] Simoncelli E.P. (1994), "Design of multi-dimensional derivative filters", IEEE Int. Conf. Image Processing, Vol. 1, pp790-793.