Computer Vision and Image Understanding **85,** 209–231 (2002)
doi:10.1006/cviu.2002.0970

Range Flow Estimation

Hagen Spies

ICG-III: Phytosphere, Research Center Jülich, 52425 Jülich, Germany
E-mail: h.spies@fz-juelich.de

Bernd Jähne

Interdisciplinary Center for Scientific Computing, University of Heidelberg, INF 368, 69120 Heidelberg, Germany

and

John L. Barron

Department of Computer Science, University of Western Ontario, London, Ontario N6A 5B7, Canada

Received March 16, 2002; accepted June 19, 2002

We discuss the computation of the instantaneous 3D displacement vector fields of deformable surfaces from sequences of range data. We give a novel version of the basic motion constraint equation that can be evaluated directly on the sensor grid. The various forms of the aperture problem encountered are investigated and the derived constraint solutions are solved in a total least squares (TLS) framework. We propose a regularization scheme to compute dense full flow fields from the sparse TLS solutions. The performance of the algorithm is analyzed quantitatively for both synthetic and real data. Finally we apply the method to compute the 3D motion field of living plant leaves. © 2002 Elsevier Science (USA)

Key Words: range data sequences; total least squares; aperture problem; regularization.

1. INTRODUCTION

The instantaneous velocity field that describes the motion of a deformable surface is denoted *range flow*. Range flow refers to a velocity field derived from sequences of range data sets. Together with the 3D structure, the range flow field can be used to study the dynamic changes of such surfaces. The same displacement vector field has also been called scene flow when computed directly from stereo image sequences [1, 2]. Using multiple

1077-3142/02 $35.00
© 2002 Elsevier Science (USA)
All rights reserved.

cameras this has been extended to additionally compute the shape and reflectance of the observed surface [3].

Most previous work on range sequence analysis seeks to estimate the sensor position with respect to a rigid environment [4–9]. The methods put forward here estimate the instantaneous velocity field of a moving deformable surface. To do so the velocity is described by a differential equation which parameters are assumed to be constant within small surface patches. This is in contrast to previously reported motion estimation on nonrigid surfaces using finite element models where the surface movement is described by a deforming mesh [10–12].

The study of range flow can be seen as an extension of 2D optical flow estimation (an approximation of the local image motion) in image sequences [13, 14]. Such image motion estimation can be feature-based, where points of interest are first segmented and then tracked through the sequence. Alternatively, it is possible to determine where a region in one image moves, based on the grayvalue information. Such area-based approaches can again be subdivided into matching and differential methods. The former approach estimates the displacement by minimizing an energy functional comprising the similarity of two candidate regions. Examples are cross-correlation, sum of squared differences, and mean absolute difference. Differential methods, on the other hand, compute that displacement vector which best satisfies a grayvalue continuity equation. In principle any optical flow method can be extended to work with range data. In this context the presented range flow estimation framework can be considered a differential technique.

The primary contributions of this work are the development of a new range flow constraint equation that can be evaluated directly on the sensor grid, the adaptation of total least squares to the problem of range flow estimation, and the introduction of a constrained regularization scheme to compute dense velocity fields from sparse TLS estimates.

Paper Organization

Section 2 discusses the fundamental constraint equation for range flow. The aperture problem encountered when this equation is used is examined in Section 2.1. We also introduce an additional constraint derived from the usually available intensity texture in Section 2.2. After the constraint equations have been introduced we show in Section 3 how the velocity field may be estimated locally within a total least squares framework. Section 4 introduces a variational regularization method to obtain a dense flow field from the sometimes sparse TLS estimates. Finally we present a number of experiments on both synthetic and real data in Section 5.

2. MOTION CONSTRAINT EQUATION

A time-varying surface may be viewed as a depth function $Z(X, Y, t)$. If the object under consideration is made up of local planar patches then this function can be expressed by its first order Taylor series expansion as

$$Z(X, Y, t) = Z_0(t) + Z_X X(t) + Z_Y Y(t), \tag{1}$$

where the Z_X, Z_Y denote the partial derivatives with respect to X and Y, respectively. Here X, Y are the local world coordinates around the point of interest. The change in depth with

time then becomes

$$\frac{\mathrm{d}Z}{\mathrm{d}t} = \frac{\partial Z_0}{\partial t} + Z_X \frac{\mathrm{d}X}{\mathrm{d}t} + Z_Y \frac{\mathrm{d}Y}{\mathrm{d}t} + X \frac{\mathrm{d}Z_X}{\mathrm{d}t} + Y \frac{\mathrm{d}Z_Y}{\mathrm{d}t}. \tag{2}$$

Under the assumption that the infinitesimal motion of the patch is a pure translation, i.e., the slope does not change ($\frac{dZ_X}{dt} = \frac{dZ_Y}{dt} = 0$), this can be written as

$$W = Z_X U + Z_Y U + Z_t. \tag{3}$$

The local displacements form the range flow field denoted by $\boldsymbol{f} = [U \;\; V \;\; W]^T = \frac{d}{dt}[X \;\; Y \;\; Z]^T$. Equation (3) is called the *range flow constraint equation* [12]. The same equation has also been termed the *elevation rate constraint equation* [5]. It is the analog of the *brightness change constraint equation* used in optical flow calculation [15]. As with image motion computation we rely on the coherence of motion to solve the correspondence problem [14]. This implies that the temporal sampling is dense enough to avoid aliasing.

Note that the above Taylor series approximation only holds on a continuous surface. At discontinuities caused by sharp edges or occlusions the constraint equation is not valid. This will cause the TLS estimate to be described in Section 3, to fail. However, for many practical applications (e.g., Section 5.4), the observed scene consists primarily of continuous surfaces.

In order to evaluate the range flow motion constraint equation (3) the derivatives of the depth function with respect to the other world coordinates have to be computed. This is not entirely straightforward for unevenly sampled data. One approach computes the desired derivatives from a local parameterized model of the surface [16, 17]. Alternatively the data may be resampled on a regular grid using a membrane [5] or thin-plate splines [9]. However, we would rather work directly on the original data, as this avoids any erroneous interpolation and avoids a computationally expensive preprocessing step.

Toward this end, we now derive a new version of the range flow constraint equation. Notice that the range sensors considered here do not yield a depth function in terms of X and Y but rather produce one data set for each of X, Y, and Z on its sampling grid ($X = X(x, y)$ etc.). Here sensor coordinates are denoted by (x, y). Note that the measured (X, Y, Z) (x, y) values are not independent of each other, because the observed 3D point has to lie on the ray of sight corresponding to the sampling position (x, y).

The three components of the range flow field are the total derivatives of the world coordinates with respect to time ($U = \frac{dX}{dt}$, etc.). Taking these derivatives yields the following equations:

$$U = \partial_x X \dot{x} + \partial_y X \dot{y} + \partial_t X, \tag{4}$$

$$V = \partial_x Y \dot{x} + \partial_y Y \dot{y} + \partial_t Y, \tag{5}$$

$$W = \partial_x Z \dot{x} + \partial_y Z \dot{y} + \partial_t Z. \tag{6}$$

Total derivatives with respect to time are indicated by a dot and partial derivatives as $\partial_x X = \frac{\partial X}{\partial x}$. As we are not interested in the rates of change in the sensor coordinate frame we eliminate $\dot{x}$ and $\dot{y}$ to obtain the range flow motion constraint expressed in sensor coordinates as

$$\frac{\partial(Z, Y)}{\partial(x, y)} U + \frac{\partial(X, Z)}{\partial(x, y)} V + \frac{\partial(Y, X)}{\partial(x, y)} W + \frac{\partial(X, Y, Z)}{\partial(x, y, t)} = 0, \tag{7}$$

where

$$\frac{\partial(Z,Y)}{\partial(x,y)} = \begin{vmatrix} \partial_x Z & \partial_x Y \\ \partial_y Z & \partial_y Y \end{vmatrix} = \partial_x Z \partial_y Y - \partial_y Z \partial_x Y \tag{8}$$

is the Jacobian of Z, Y with respect to x, y. Notice that the Jacobians are readily computed from the derivatives of X, Y, Z in the sensor frame using derivative kernels. Thus the new constraint can be evaluated entirely on the sensor grid by means of convolutions. This makes it possible to draw on well-established linear filter theory and can be implemented very efficiently.

Equation (7) poses the general constraint equation independent of a particular sensor. Many sensors have aligned world and sensor coordinate systems ($\partial_y X = \partial_x Y = 0$). This simplifies the range flow constraint equation considerably:

$$\begin{aligned}(\partial_y Y \partial_x Z)U + (\partial_x X \partial_y Z)V - (\partial_x X \partial_y Y)W \\ + (\partial_x X \partial_y Y \partial_t Z - \partial_x X \partial_t Y \partial_y Z + \partial_t X \partial_y Y \partial_x Z) = 0.\end{aligned} \tag{9}$$

Not surprisingly, dividing through $(\partial_x X \partial_y Y)$ yields the initial constraint equation (3). But notice that the partial derivative of Z with respect to time depends on the parameterization:

$$(\partial_t Z)_{XY} = (\partial_t Z)_{xy} + (\partial_X Z)_{XY}(\partial_t X)_{xy} + (\partial_X Z)_{XY}(\partial_t Y)_{xy}. \tag{10}$$

Here we still make the assumption of aligned sensor and world coordinate systems. In some cases one can further assume orthogonal projection onto the sensor, for instance when a telecentric lens is used. This implies $\partial_x X = \Delta x$, $\partial_y Y = \Delta y$, and $\partial_t X = \partial_t Y = 0$, where Δx, Δy are constants. In this case the constraint equation simply becomes

$$\frac{\partial_x Z}{\Delta x} U + \frac{\partial_y Z}{\Delta y} V - W + \partial_t Z = 0. \tag{11}$$

2.1. The Aperture Problem

Unfortunately (3) or (7) poses only one constraint in three unknowns; this may be attributed to our initial assumption of locally planar surface patches. On a plane only the movement perpendicular to the plane may be observed. This is a three-dimensional version of the aperture problem known from optical flow [18]. Just as the optical flow constraint equation describes a line in 2D velocity space, so does the range flow constraint equation (3) describe a plane in (U, V, W)-space with surface normal $[Z_X \;\; Z_Y \;\; -1]^T$. The best solution that can be achieved will be the vector with minimum norm, between the origin and the constraint plane. This solution defines the *raw normal flow*:

$$\boldsymbol{f}_r = \frac{-Z_t}{Z_X^2 + Z_Y^2 + 1} \begin{bmatrix} Z_X \\ Z_Y \\ -1 \end{bmatrix}. \tag{12}$$

In the following the linear dependencies that may occur when the range flow constraint equations are pooled over a certain region are examined. Here we only discuss range flow: however, we note that for volumetric flow from time-varying 3D data such as CT scans the

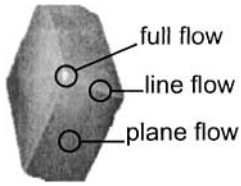

FIG. 1. Example range data from a cube that illustrates the three different types of neighborhoods encountered in three-dimensional data.

same problems occur. The three possible types of neighborhoods are illustrated in Fig. 1. On corner- or point-like structures clearly all three components of the movement can be determined even locally; such an estimate will in the following be called *full flow*. Any movement along edges or linear structures cannot be resolved by a local analysis, but only the velocity in the perpendicular plane, which we call *line flow*. On planar surfaces any velocity within the plane is not detectable. Only the component perpendicular to the surface can be determined, which will be denoted *plane flow*. The implications of the three region types for the constraint equations are discussed next, noting that they define planes in velocity space.

As we are dealing with three-dimensional flow, at least three mutually distinct, i.e., non-parallel, planes are needed to combine the constraint equations from a local region to yield full 3D flow. If enough such linearly independent constraint equations are available in the considered neighborhood the full velocity vector is readily computed by the intersection of the constraint planes as illustrated in Fig. 2a.

Linear structures such as intersecting planes result in two distinct classes of constraint planes in the examined aperture; see Fig. 2b. The point on the common line closest to the origin gives the appropriate line flow. This line flow lies in the plane perpendicular to the linear structure and only the motion along the direction of the structure cannot be resolved.

Obviously the considered neighborhood may be made up of a single plane. Then all constraint equations are essentially the same and we only have one averaged constraint plane; see Fig. 2c. The point on this plane closest to the origin defines the velocity normal to the planar surface. This plane flow can be obtained by using averaged derivative values in (12). Note that by considering an entire neighborhood such a plane flow estimate is less sensitive to noise than the raw normal flow computed via (12).

Section 3 describes how a local estimate is obtained by means of a total least squares technique. In Section 3.1 it is shown how the described linear dependencies can be automatically detected within this framework. Last, Section 3.2 shows how to compute the appropriate

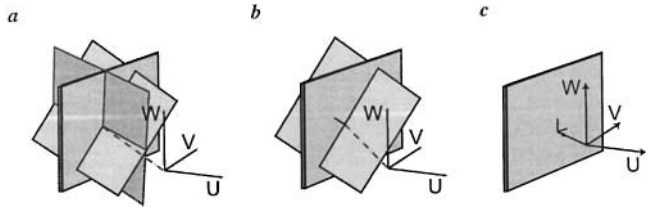

FIG. 2. Illustration of the independent constraint equations present in a neighborhood for (a) full flow, (b) line flow, and (c) plane flow.

normal flows. An alternative least squares solution for range flow and the appropriate normal flows has been described elsewhere [19, 20].

2.2. Intensity Constraint

Apart from the 3D structure information discussed so far, optical range sensors also return an intensity value of the observed surface. Clearly the information thus available should be exploited as well. In order to derive another constraint equation it is reasonable to assume that the intensity of an observed point remains the same for moderate depth changes. Thus, as for optical flow, we attribute all changes in intensity to motion in the horizontal plane. This yields another constraint equation:

$$\frac{\mathrm{d}I}{\mathrm{d}t} = \partial_x I\,\dot{x} + \partial_y I\,\dot{y} + \partial_t I = 0. \tag{13}$$

Combined with Eqs. (4) and (5) we obtain an additional constraint on U and V:

$$\frac{\partial(I, Y)}{\partial(x, y)}U + \frac{\partial(X, I)}{\partial(x, y)}V + \frac{\partial(X, Y, I)}{\partial(x, y, t)} = 0. \tag{14}$$

As before this can also be written in the alternative form.

$$0 = I_X U + I_Y U + I_t. \tag{15}$$

Because we assumed the intensity is not altered by changes in the distance we cannot infer any information about the vertical motion W. In an actual range flow estimation algorithm we seek a solution that fulfills both (14) and (7). Section 3.3 shows how this can be achieved. Apart from making an estimate more robust by using all available data, the additional intensity constraint often allows an estimate of full range flow in places where the range constraint equation alone is not adequate [21].

3. LOCAL TLS SOLUTION

The TLS solution presented here is an extension of the *structure tensor* algorithm for optical flow estimation [14, 22]. The method may also be viewed as a special case of a more general technique for parameter estimation in image sequences [23].

Assuming constant flow in a region containing n pixels we have n equations (7). These equations can be written in the form $\boldsymbol{d}^T \tilde{\boldsymbol{f}} = 0$, where

$$\boldsymbol{d} = \left[\frac{\partial(Z, Y)}{\partial(x, y)} \frac{\partial(X, Z)}{\partial(x, y)} \frac{\partial(Y, X)}{\partial(x, y)} \frac{\partial(X, Y, Z)}{\partial(x, y, t)}\right]^T \quad \text{and} \quad \tilde{\boldsymbol{f}} = [U\,V\,W\,1]^T. \tag{16}$$

Because Eq. (7) gives only one constraint for every pixel we need to make further assumptions in order to solve for the complete parameter field. For least-squares estimation we pool the constraints over a local neighborhood and assume the parameters to be constant within this area. Another possible assumption is made by variational approaches that impose smoothness constraints; such a method has been described in [21, 24]. The assumption of locally constant parameters leads to the following minimization problem:

$$\boldsymbol{p} = \arg\min \int_{-\infty}^{\infty} w(\boldsymbol{x} - \boldsymbol{x}', t - t')(\boldsymbol{d}^T \boldsymbol{p})^2 \,\mathrm{d}\boldsymbol{x}'\,\mathrm{d}t' \text{ subject to } \boldsymbol{p}^T \boldsymbol{p} = 1. \tag{17}$$

Here $w(\boldsymbol{x}-\boldsymbol{x}', t-t')$ is a weighting function that defines the spatiotemporal neighborhood where the parameters are estimated. Because we require $\|\boldsymbol{p}\| = 1$ the trivial solution is avoided. This later requirement can be incorporated by means of a Lagrangian multiplier to yield the unconstrained energy functional

$$E = \int_{-\infty}^{\infty} w(\boldsymbol{x}-\boldsymbol{x}', t-t') \left[(\boldsymbol{d}^T \boldsymbol{p})^2 + \lambda \left(1 - \sum_{i=1}^{n} p_i^2 \right) \right] \mathrm{d}\boldsymbol{x}' \, \mathrm{d}t'. \tag{18}$$

It is known from basic calculus that this energy functional reaches a minimum when the derivatives with respect to all variables vanish. These derivatives take the following form:

$$\frac{\partial E}{\partial p_i} = 2 \int_{-\infty}^{\infty} w(\boldsymbol{x}-\boldsymbol{x}', t-t')[d_i(\boldsymbol{d}^T \boldsymbol{p}) - \lambda p_i] \, \mathrm{d}\boldsymbol{x}' \, \mathrm{d}t' \doteq 0, \quad \forall i = 1 \ldots n. \tag{19}$$

We are assuming a locally constant parameter field $\boldsymbol{p}$; hence the p_i can be taken out of the integral:

$$\left(\int d_i d_1 \right) p_1 + \cdots + \left(\int d_i d_n \right) p_n = \lambda p_i; \quad \forall i = 1 \ldots n. \tag{20}$$

On the right hand side of Eq. (19) we assume the weight function to be normalized; i.e., $\int w = 1$. Then these n equations can be written in matrix form as

$$\boldsymbol{J}\boldsymbol{p} = \lambda \boldsymbol{p} \quad \text{with } \boldsymbol{J} = \int_{-\infty}^{\infty} w(\boldsymbol{x}-\boldsymbol{x}', t-t')(\boldsymbol{d}\boldsymbol{d}^T) \, d\boldsymbol{x}' \, dt'. \tag{21}$$

This is an eigenvalue equation of the extended structure tensor $\boldsymbol{J}$. For each of the n eigenvectors we get an extremum of the energy functional E. In a discrete implementation, the components of $\boldsymbol{J}$ can be readily computed using standard image processing operations applied to the data vector d, $\boldsymbol{J} = \boldsymbol{B} * (\boldsymbol{d}\boldsymbol{d}^T)$, where $*$ denotes convolution. $\boldsymbol{B}$ is an averaging filter containing the weights w, typically Binomial or Box. $\boldsymbol{J}$ contains all the necessary information about the local spatiotemporal structure of the data; it is an extension of the structure tensor encountered in optical flow computation [14].

The energy (17) to be minimized can be written as follows:

$$\int_{-\infty}^{\infty} w(\boldsymbol{d}^T \boldsymbol{p})^2 = \boldsymbol{p}^T \boldsymbol{J} \boldsymbol{p} = \lambda. \tag{22}$$

Because $\boldsymbol{p}$ is normalized this energy simply becomes the corresponding eigenvalue for each solution. It follows that (17) is minimized by the eigenvector $\hat{e}_4$ corresponding to the smallest eigenvalue λ_4 of $\boldsymbol{J}$.

This minimization problem (17) is also known as the orthogonal l_2 approximation problem and the solution coincides with the TLS solution up to a scaling factor [25]. As $\boldsymbol{J}$ is real and symmetric, the eigenvalues and -vectors can easily be computed using Jacobi rotations [26]. The desired range flow is then given by

$$\boldsymbol{f}_f = \frac{1}{e_{44}} \begin{bmatrix} e_{14} \\ e_{24} \\ e_{34} \end{bmatrix}. \tag{23}$$

In order to save execution time and increase overall flow accuracy we only compute range flow where the trace of the tensor exceeds a threshold τ_1. This eliminates regions with insufficient magnitude of the gradient. The regularization step described in Section 4 subsequently closes these holes.

3.1. Confidence Measures

In the above we are really fitting a local constant flow model to the data. The smallest eigenvalue λ_4 directly measures the quality of this fit (22). In particular, at motion discontinuities, the data cannot be described by a single flow and the fit fails. This is also the case for pure noise without any coherent motion. To quantify this we introduce a confidence measure:

$$\omega = \begin{cases} 0 & \text{if } \lambda_4 > \tau_2 \text{ or } tr(\boldsymbol{D}) < \tau_1 \\ \left(\frac{\tau_2 - \lambda_4}{\tau_2 + \lambda_4}\right)^2 & \text{otherwise .} \end{cases} \tag{24}$$

Here τ_2 is used to determine whether an eigenvalue is considered nonvanishing. To choose this threshold, note that ideally the smallest eigenvalue will become $\lambda_4 \approx \langle \sigma^2 \rangle$, where σ^2 is the noise level in $\boldsymbol{d}$ and $\langle \cdot \rangle$ denotes the local average computed using the filter $\boldsymbol{B}$. On this basis a threshold can be chosen, for example $\tau_2 = 3 \langle \sigma^2 \rangle$. Of course there still is the need for an informed guess about the noise variance. In practice one will often choose τ_2 empirically.

Even a high confidence value by no means ensures that all the parameters can be estimated independent of each other. The reason is that more than one eigenvalue may be close to zero ($<\tau_2$) and we can no longer uniquely pick an eigenvector as a solution. More generally, any vector in the nullspace of $\boldsymbol{J}$ yields a possible solution. This is the aperture problem revisited; see Section 2.1. Nevertheless, we can compute a sensible estimate, as shown in Section 3.2. Thus, we want to compute a type metric that indicated how well the dimensionality of the nullspace of $\boldsymbol{J}$ has been determined. This can be done by examining how much the first relevant eigenvalue λ_p is above the threshold. We define

$$\omega_t = \left(\frac{\lambda_p - \tau_2}{\lambda_p}\right)^2 \tag{25}$$

as a normalized measure for each encountered type (plane flow $p = 1$, line flow $p = 2$, and full flow $p = 3$).

3.2. Normal Flows

As pointed out above, any vector in the nullspace of $\boldsymbol{J}$ is a possible solution. However, of all these solutions only that with minimum norm is actually observable. Any other motion components do not result in detectable changes in the measured data. Yet it is important to keep in mind that the "real" solution can be any vector in the nullspace. It is precisely this degeneracy that motivates the constrained interpolation algorithm presented in Section 4. But first we discuss how to obtain the minimum norm solutions. Towards this end the possible solutions are expressed as linear combinations of the relevant eigenvectors and the result with minimal Euclidean norm is chosen. Appendix A gives the details.

As discussed in Section 2.1 we encounter two types of normal flows for range flow. On linear structures the motion along the line cannot be resolved. This results in two vanishing

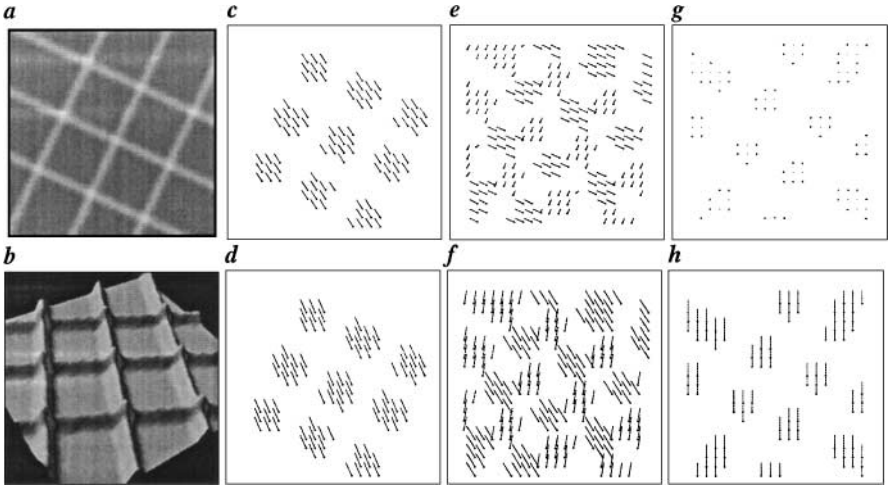

FIG. 3. Example flow types for range flow: (a) synthetic depth map, (b) rendered. X–Y components of the estimated flow fields, (c) full flow, (e) line flow, and (g) plane flow, and X–Z components of the estimated flow field, (d) full flow, (f) line flow, and (h) plane flow.

eigenvalues of the structure tensor. We name the corresponding minimum norm velocity *line flow* which is computed from (A.10) as follows:

$$\boldsymbol{f}_l = \frac{1}{1 - e_{14}^2 - e_{24}^2} \left[e_{14} \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \end{bmatrix} + e_{24} \begin{bmatrix} e_{21} \\ e_{22} \\ e_{23} \end{bmatrix} \right]. \tag{26}$$

When we encounter planar structures only the movement perpendicular to the plane can be resolved. Hence all but one eigenvalue falls below τ_2 and the *plane flow* becomes

$$\boldsymbol{f}_p = \frac{e_{14}}{1 - e_{14}^2} \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \end{bmatrix} = \frac{e_{14}}{e_{11}^2 + e_{12}^2 + e_{13}^2} \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \end{bmatrix}. \tag{27}$$

On pointlike structures, motion in all directions can be measured and the resulting *full flow* is computed as usual from the eigenvector to the single vanishing eigenvalue λ_4:

$$\boldsymbol{f}_f = \frac{1}{e_{44}} \begin{bmatrix} e_{41} \\ e_{42} \\ e_{43} \end{bmatrix}. \tag{28}$$

Figure 3 shows a synthetic example that nicely illustrates all three kinds of encountered flow types.

3.3. Incorporating the Intensity

As illustrated in Section 2.2 the intensity yields another, constraint on the X- and Y-components of the sought range flow. In order to incorporate this constraint we have to

insert zero at the appropriate place. Then (14) translates into $\boldsymbol{d}_i^T \tilde{\boldsymbol{f}}$ with

$$\boldsymbol{d}_i = \left[\frac{\partial(I, Y)}{\partial(x, y)} \frac{\partial(X, I)}{\partial(x, y)} 0 \frac{\partial(X, Y, I)}{\partial(x, y, t)}\right]^T \quad \text{and} \quad \tilde{\boldsymbol{f}} = [U\,V\,W\,1]^T. \tag{29}$$

The combined energy to be minimized can then be written (see Eq. (17)) as,

$$\boldsymbol{p} = \arg\min \int_{-\infty}^{\infty} w(\boldsymbol{x} - \boldsymbol{x}', t - t')\left[(\boldsymbol{d}^T \boldsymbol{p})^2 + \beta\left(\boldsymbol{d}_i^T \boldsymbol{p}\right)^2\right] \mathrm{d}\boldsymbol{x}'\,\mathrm{d}t'. \tag{30}$$

The two data channels are weighted using a constant β. This is useful, for instance, when they have different signal-to-noise ratios. We scale the intensity to have the same mean and variance as the depth data before combining the two channels. Incorporating the additional constraint $\|\boldsymbol{p}\| = 1$ leads to a combined structure tensor that is simply the weighted sum of the two tensors on the depth and intensity channels,

$$\boldsymbol{J}' = \boldsymbol{J} + \beta \boldsymbol{J}_i \quad \text{with } \boldsymbol{J}_i = \int_{-\infty}^{\infty} w(\boldsymbol{x} - \boldsymbol{x}', t - t')\left(\boldsymbol{d}_i \boldsymbol{d}_i^T\right) \mathrm{d}\boldsymbol{x}'\,\mathrm{d}t', \tag{31}$$

and $\boldsymbol{J}$ as given by (21). The above discussion remains true for the combined tensor $\boldsymbol{J}'$.

The use of the intensity has two advantages. First, more data points are used for the local estimation, which increases the accuracy. Second, and more important, through the use of intensity as well as depth information it is often easier to resolve the aperture problem. For example, the presence of a dot-oriented texture on a planar surface may enable the computation of full flow where otherwise only plane flow could be estimated.

To illustrate the beneficial effect of additional intensity information we use the synthetic sequence shown in Fig. 4. The correct movement is $[0.66, -0.46, 0.34]^T$. Using depth maps alone local TLS estimation gives only sparse full flow, Fig. 4c. The flat surface only allows the computation of plane flow perpendicular to it. On the lines any movement in the direction of the lines can not be resolved. When the linear intensity pattern (Fig. 4b) is taken into consideration the displacements along the lines can now be computed as well. Hence the full flow density increases from 7% to 38%, as shown in Fig. 4d. Clearly this would not be the case if intensity and depth lines coincided. On the planar surface the intensity allows computation normal to the intensity lines, yet one direction remains ambiguous. Both the relative error (47) and the directional error (48) are very low on this ideal test case ($<1\%$ and $<1^\circ$).

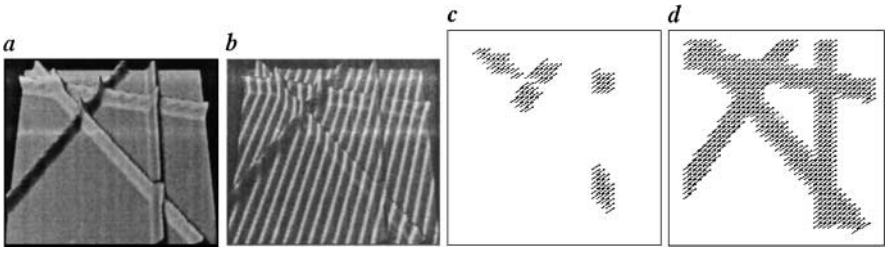

FIG. 4. Rendered synthetic data: (a) depth only and (b) with intensity texture. X–Y range flow: (c) depth only and (d) using the intensity as well.

4. REGULARIZATION

Even the use of the additional intensity data does not guarantee a full flow estimate everywhere. There are regions where still only one of the normal flows may be recovered, sometimes the estimation fails, and occasionally there is not enough variation in the data for a sensible flow calculation. We will now present a regularization scheme that makes it possible to close these remaining holes while taking the information from the previous TLS estimation into account.

We seek to estimate a dense and smooth flow field, $v = [U \;\; V \;\; W]^T$. In places where flow estimates from the above TLS algorithm exist we denote them $\boldsymbol{f}$, computed from one of Eqs. (26), (27), or (28) as appropriate. From the structure of the TLS solution given by (A.10) we can use the reduced eigenvectors as a, not necessarily orthogonal, basis for the desired solution:

$$\boldsymbol{b}_i = \frac{1}{\sum_{k=1}^{3} e_{ki}^2} \begin{bmatrix} e_{1i} \\ e_{2i} \\ e_{3i} \end{bmatrix} \quad i = 1, 2, 3. \tag{32}$$

Using this notation we define a projection matrix which projects onto the subspace that was determined by our TLS algorithm:

$$\boldsymbol{P} = \bar{\boldsymbol{B}}_p \bar{\boldsymbol{B}}_p^T \quad \text{where} \quad \lambda_p > \lambda_{p+1} \approx \cdots \approx \lambda_4 \approx 0, \tag{33}$$

$$\bar{\boldsymbol{B}}_p = [\hat{b}_1 \cdots \hat{b}_p]. \tag{34}$$

Each estimated flow vector $\boldsymbol{f} = \boldsymbol{f}_{f,p,l}$ constrains the solution within this subspace. We therefore require the regularized solution to be close in a least-squares sense,

$$\omega(\boldsymbol{P}\boldsymbol{v} - \boldsymbol{f})^2 \to \min, \tag{35}$$

where ω, given by Eq. (24), captures the confidence of the TLS solution. At locations where no solution has been computed, obviously no such data term exists. This is readily accounted for by setting the confidence values ω to zero.

To ensure smoothly varying parameters we use a membrane smoothness term,

$$\sum_{i=1}^{3} (\nabla v_i)^2 \to \min. \tag{36}$$

Obviously the use of this simple membrane model is only justified if we have already segmented the data into differently moving objects. If no such segmentation were available more elaborate schemes would have to be considered. The above smoothness term usually considers only spatial neighborhoods ($\nabla = [\partial x, \partial y]^T$); however, this is easily extended to enforce temporal smoothness as well ($\nabla = [\partial x, \partial y, \partial t]^T$).

Combining the data (35) and smoothness (36) terms in the considered area A yields the following minimization problem:

$$\int_A \underbrace{\left\{ \omega(\boldsymbol{P}\boldsymbol{v} - \boldsymbol{f})^2 + \alpha \sum_{i=1}^{3} (\nabla v_i)^2 \right\}}_{h(v)} \mathrm{d}\boldsymbol{r} \to \min. \tag{37}$$

The overall smoothness can be regulated by the positive nonzero constant α. The minimum of (37) is achieved when the Euler–Lagrange equations are satisfied:

$$\frac{\partial h}{\partial v_i} - \frac{\mathrm{d}}{\mathrm{d}x}\frac{\partial h}{\partial (v_i)_x} - \frac{\mathrm{d}}{\mathrm{d}y}\frac{\partial h}{\partial (v_i)_y} = 0; \quad i = 1, 2, 3. \tag{38}$$

If an extension in the temporal domain is anticipated another term $-\frac{d}{dt}\frac{\partial h}{\partial (v_i)_t}$ will have to be added. Subscripts x, y, t denote partial differentiation. Using vector notation we write the Euler–Lagrange equations as follows:

$$\frac{\partial h}{\partial \boldsymbol{v}} - \frac{d}{dx}\frac{\partial h}{\partial (\boldsymbol{v}_x)} - \frac{d}{dy}\frac{\partial h}{\partial (\boldsymbol{v}_y)} = 0. \tag{39}$$

Computing the derivatives yields

$$2\omega \boldsymbol{P}(\boldsymbol{P}\boldsymbol{v} - \boldsymbol{f}) - 2\alpha\left[\frac{d}{dx}(\boldsymbol{v}_x) + \frac{d}{dy}(\boldsymbol{v}_y)\right] = 0. \tag{40}$$

Introducing the Laplacian $\Delta \boldsymbol{v} = \boldsymbol{v}_{xx} + \boldsymbol{v}_{yy}$ we get

$$\omega \boldsymbol{P}\boldsymbol{v} - \omega \boldsymbol{P}\boldsymbol{f} - \alpha \Delta \boldsymbol{v} = 0, \tag{41}$$

where the idempotence of the projection matrix $\boldsymbol{P}\boldsymbol{P} = \boldsymbol{P}$ is used. The Laplacian can be approximated as $\Delta \boldsymbol{v} = \bar{\boldsymbol{v}} - \boldsymbol{v}$, where $\bar{\boldsymbol{v}}$ denotes a local average [18]. Using this approximation we arrive at

$$(\omega \boldsymbol{P} + \alpha \mathbb{1})\,\boldsymbol{v} = \alpha \bar{\boldsymbol{v}} + \omega \boldsymbol{P}\boldsymbol{f}. \tag{42}$$

This makes possible an iterative solution to the minimization problem. We introduce $\boldsymbol{A} = \omega \boldsymbol{P} + \alpha \mathbb{1}$ and get an update $\boldsymbol{v}^{k+1}$ from the solution at step k:

$$\boldsymbol{v}^{k+1} = \alpha \boldsymbol{A}^{-1}\bar{\boldsymbol{v}}^k + \omega \boldsymbol{A}^{-1}\boldsymbol{P}\boldsymbol{f}. \tag{43}$$

Because of the convex energy functional convergence is guaranteed; thus we could use zero for initialization. However, here we are using the TLS solution, as this slightly improves the rate of convergence. Examining $\boldsymbol{A}$, we find

$$\boldsymbol{A}^{-1} = \alpha^{-1}\left(\mathbb{1} - \frac{\omega}{\alpha + \omega}\boldsymbol{P}\right). \tag{44}$$

To verify that $\boldsymbol{A}^{-1}\boldsymbol{A} = \mathbb{1}$ only the idempotence of the projection matrix is needed. We can thus examine the iteration process in more detail. Inserting (44) into (43) yields

$$\boldsymbol{v}^{k+1} = \bar{\boldsymbol{v}}^k - \frac{\omega}{\alpha + \omega}\boldsymbol{P}\bar{\boldsymbol{v}}^k + \frac{\omega}{\alpha + \omega}\boldsymbol{P}\boldsymbol{f}, \tag{45}$$

where we also used the idempotence of $\boldsymbol{P}$. Using $\mathbb{1} = \boldsymbol{P} + \boldsymbol{P}^{\perp}$ we can separate the two subspaces:

$$\boldsymbol{v}^{k+1} = \boldsymbol{P}^{\perp}\bar{\boldsymbol{v}}^k + \frac{1}{\alpha + \omega}\boldsymbol{P}(\alpha \bar{\boldsymbol{v}}^k + \omega \boldsymbol{f}). \tag{46}$$

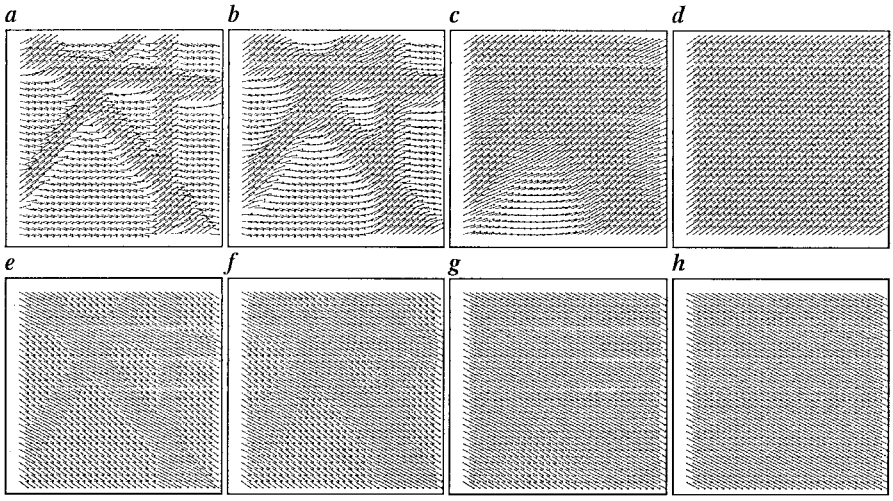

FIG. 5. Example of regularization for the range flow field computed on the data given in Fig. 4. X–Y-flow filed after (a) 10, (b) 100, (c) 1000, and (d) 10,000 iterations. X–Z-flow field after (e) 10, (f)100, (g) 1000, and (h) 10,000 iterations.

In the orthogonal subspace, where no initial TLS solution is available, the local average is used. In the subspace determined by the TLS solution the iterative update is given by a weighted mean of the local average and the originally available solution. The weights are determined by the regularization constant α and the confidence measure ω. Notice that the choice of α only regulates the amount of smoothing in the TLS subspace and has no effect in the orthogonal subspace.

The effect of this regularization for a range flow field is illustrated in Fig. 5 where the algorithm is applied to the data given in Fig. 4. Here we used a regularization parameter, $\alpha = 1$. In the beginning the flow field is dominated by the normal flows. With increasing iterations the full flow information spreads out. However, due to the large regions without initial full flow estimation it takes a very long time to reach convergence. In this example a 5×5 Box filter is used to compute the average.

5. EXPERIMENTS

This section first examines the quantitative performance of the proposed algorithm on synthetic and real test data. Then examples from a real scientific application are given.

5.1. Error Measure

In order to quantify the accuracy of the estimated range flow, the following error measures are used. Let the correct range flow be denoted as $\boldsymbol{f}_c$ and the estimated flow as $\boldsymbol{f}_e$. The first error measure describes the *relative error* in the velocity magnitude:

$$E_r = \frac{|(|\boldsymbol{f}_c| - |\boldsymbol{f}_e|)|}{|\boldsymbol{f}_c|} \cdot 100\,[\%]. \tag{47}$$

Because E_r measures only the difference between the estimated and the correct velocity magnitude, no deviation from the correct direction is captured. Therefore we use the

directional error as a second error measure:

$$E_d = \arccos\left(\frac{\boldsymbol{f}_c \cdot \boldsymbol{f}_e}{|\boldsymbol{f}_c||\boldsymbol{f}_e|}\right)[^\circ]. \tag{48}$$

The value of E_d directly gives the angle in 3D between the correct velocity vector and the estimated vector and thus describes how accurately the correct direction has been recovered. Apart from the averaged error values we report their standard deviation to determine what range of error values can be expected.

5.2. Synthetic Test Data

To generate synthetic range data we model a structured light sensor with a focal length of $f = 12$ mm, a pixel size of $7.4 \times 7.4\,\mu\text{m}^2$, and 256×256 sensor elements. This resembles a sensor used to study plant leaves; see Section 5.4. To eliminate all border effects, especially for the regularized solution, we only use the inner 200×200 pixel to compute averaged error values.

The first test sequence contains a synthetic plane at a viewing distance of 300 mm and typically an angle between the surface normal and the Z-axis of $\vartheta = 5^\circ$. The intensity textures consists of a sinusoidal plaid pattern with a wavelength of 1 mm in both directions.

The second type of synthetic data uses a sphere with a radius of 300 mm with its center initially placed 700 mm away from the camera. The intensity on the sphere is described as a sinusoidal plaid in terms of the spherical angles of the surface point:

$$I = \begin{cases} o & \text{if } \theta < 0.5^\circ \\ o + a\sin\left(\frac{2\pi\theta}{\lambda_\theta}\right) + a\sin\left(\frac{2\pi\phi}{\lambda_\phi}\right) & \text{otherwise .} \end{cases} \tag{49}$$

In the following we choose wavelengths of $\lambda_\theta = 1^\circ$ and $\lambda_\phi = 30^\circ$. Figure 6 shows the computed range data using an offset $o = 100$ and an amplitude $a = 50$.

Here we are only looking at the full flow estimation performance using the simulated structured light sensor; other results for normal flows can be found in [27]. In the following the intensity data is always incorporated using $\beta = 1$ (Section 3.3). How does noise in the four data channels influences the accuracy of the recovered velocity? The results for a translation in the X-direction (only U is nonzero) on the plane tilted by $\vartheta = 5^\circ$ are shown in Fig. 7. The results for other movements are practically identical and not shown here. For both the relative magnitude error and the directional error we observe the same qualitative behavior. The range flow estimation fails for movements above 1 mm/frame. This is due to aliasing and thus dependent on the data. In the considered case the intensity structure limits the maximum velocity.

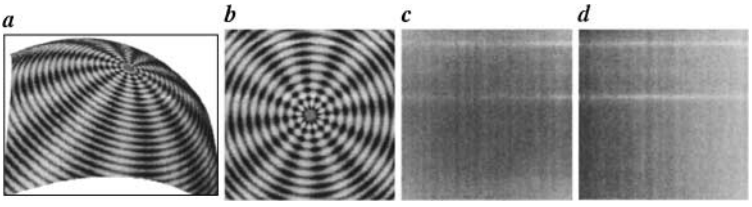

FIG. 6. Synthetic sphere: (a) texture-mapped range data, (b) intensity data, (c) depth (Z) data, and (d) X data.

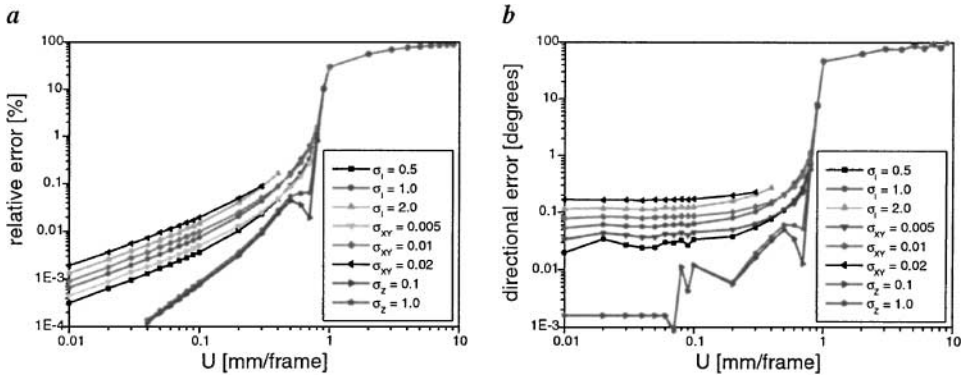

FIG. 7. Error depending on noise in the individual data channels: (a) relative magnitude and (b) direction error for movements in X direction.

We find that the results do not depend very much on the noise in the Z data but are much more influenced by the noise in the intensity and X, Y coordinates. In particular, we see that a rise of σ_{xy} from 0.005 to 0.02 mm decreases the estimation accuracy by almost one order of magnitude. This indicates that on more or less planar data, where most scenes can be considered to be locally planar, it is not the depth accuracy but that in the other coordinates which limits the velocity estimation. In the following we will examine the performance for three typical noise situations:

Name	$\sigma_X = \sigma_Y$	σ_Z	σ_I
N1	0.005	0.05	0.5
N2	0.01	0.1	1.0
N3	0.02	0.2	2.0

These are sensible choices for typical range sensors with a viewing distance around 0.5 m [28, 29]. In Fig. 8 the dependency of the relative error on four types of movements are given for the above noise models. For a translation in the X-direction we find the same behavior as above with similar error values even though there is noise on all data channels now.

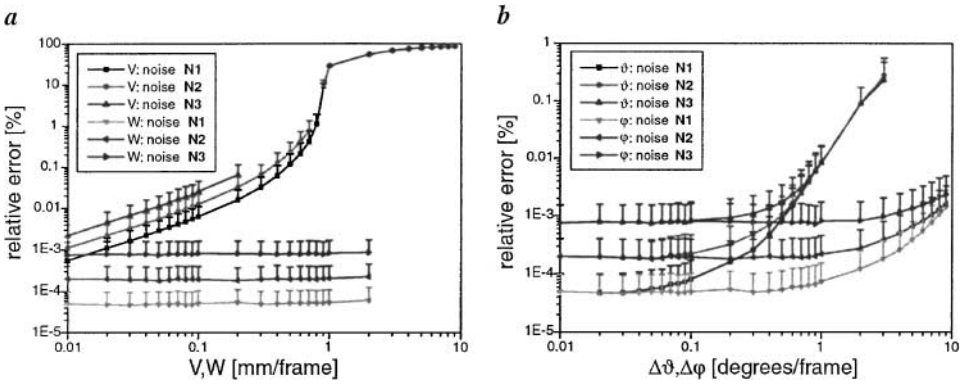

FIG. 8. Relative magnitude error for different kinds of motion: (a) translation in Y- or Z-direction, (b) rotation of the plane by ϕ and by ϑ.

The accuracy in the estimation of the Z-movement does not decrease for higher velocities until 2 mm/frame are reached. For even greater velocities no range flow is estimated as the confidence value drops to zero. Unfortunately this is not the case for high X-velocities which are computed wrongly.

Another interesting type of motion is a rotation of the plane. This can be achieved by changing the spherical angles of the surface normal, resulting in a nonuniform velocity field. The relative error for an increasing rotational velocity is given in Fig. 8b. For small changes in the angles the resulting flow field is calculated very accurately. This remains true for rotations around the Z-axis ($\Delta\phi$) even for a rotation up to 10°/frame. However, for a change in ϑ, which is the angle between the surface normal and the Z-axis, we see that the accuracy quickly deteriorates for changes above 1°/frame. This implies that a fast tilt of the observed surface cannot be captured accurately using this method, that is, if the frame rate cannot be increased accordingly.

We proceed by looking at the range flow accuracy on the data made up of a synthetic sphere. The relative and directional error for a translation in the X- and Z-directions are given in Fig. 9. We obtain the same qualitative results as for the plane test data (Fig. 8a) but on this data the error is almost one order of magnitude higher. Yet for motions smaller than 1 mm/frame the error is still below 1% and 1°, respectively.

In order to asses the algorithms performance for nonrigid motions we use two test cases. In the first example the sphere is expanded by increasing the radius. For the second example we expand the intensity pattern on the tilted plane. The resulting relative error in the TLS range flow estimation for the expanding sphere is shown in Fig. 10a. We find very low errors for an expansion of up to 1%/frame. However, already for 2%/frame the encountered speed is so high that the algorithm does not compute any flow any more. For an oriented plane with $\vartheta = 15°$ the accuracy of the computed flow is examined in Fig. 10b. We see that the error increases for higher expansion rates. Yet even for very high expansion rates, up to 10%/frame, we achieve a relative error below 1% and a directional error below 1° (not shown here). However, note that for the highest noise level (**N3**) no full flow is computed for growth rates exceeding 2%/frame.

The results of TLS estimation on the synthetic data are already very good indeed. Therefore we only examine the regularization results for the example of the moving sphere to see whether there are further improvements in the accuracy. The relative magnitude error is

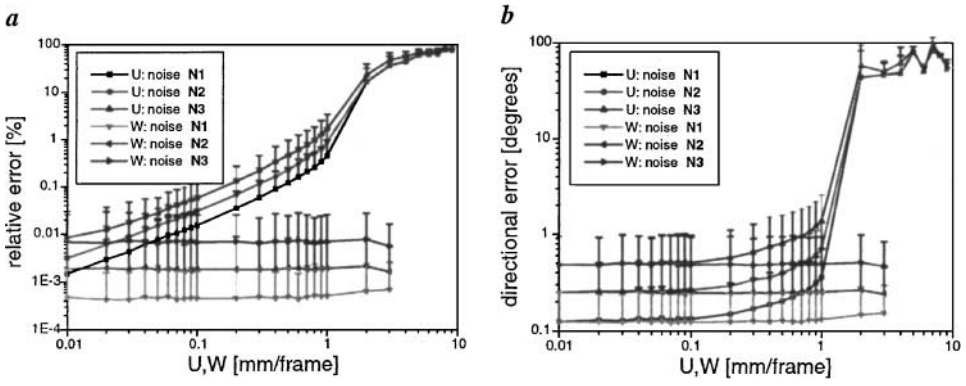

FIG. 9. Error in the TLS range flow on the sphere depending on the velocity in the X- and Z-directions: (a) relative error and (b) directional error.

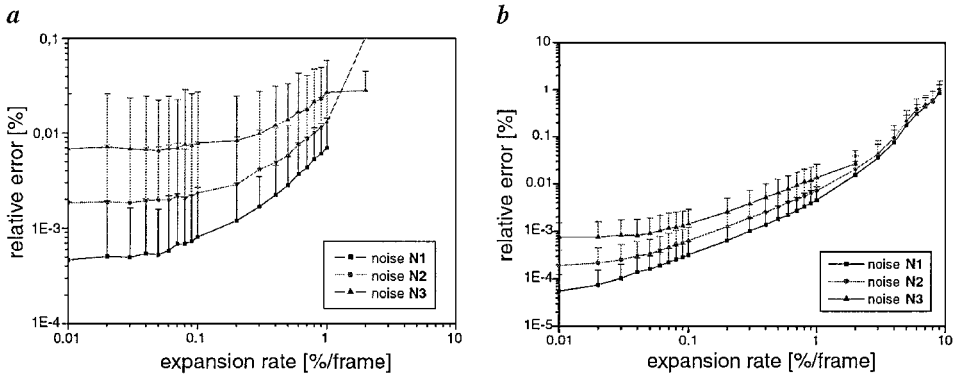

FIG. 10. Relative error in the TLS range flow for nonrigid motion: (a) expanding sphere and (b) expansion on a tilted plane.

given in Fig. 11a and the directional error in Fig. 11b. Comparing the results to those found using only TLS (Fig. 9) we see that both error values drop by about one order of magnitude. This can be attributed to the smoothing accompanied by the regularization. Here and in the following we used $\alpha = 10$ and stopped after 100 iterations.

In order to convert the presented results to another setup it is important to realize that the limiting factor is the displacement in units of the sensor grid. To a first approximation this displacement follows from perspective projection of the horizontal movement ΔX to be $\Delta x = \frac{f}{p}\frac{\Delta X}{Z}$, where p denotes the pixel size. To be on the safe side the displacement should not exceed 1 pixel/frame [14]. For the given example a displacement of 1 pixel/frame corresponds to $\Delta X = 0.19$ mm at $Z = 300$ mm and $\Delta X = 0.25$ mm at $Z = 400$ mm. As shown above, vertical displacements can be considerably larger.

5.3. Real Test Data

The results from synthetic data can only be transferred to a certain degree to real data. Here we examine a few real test sequences. In all cases the observed objects are moved using linear positioners. Thus we can only generate pure translational movements without

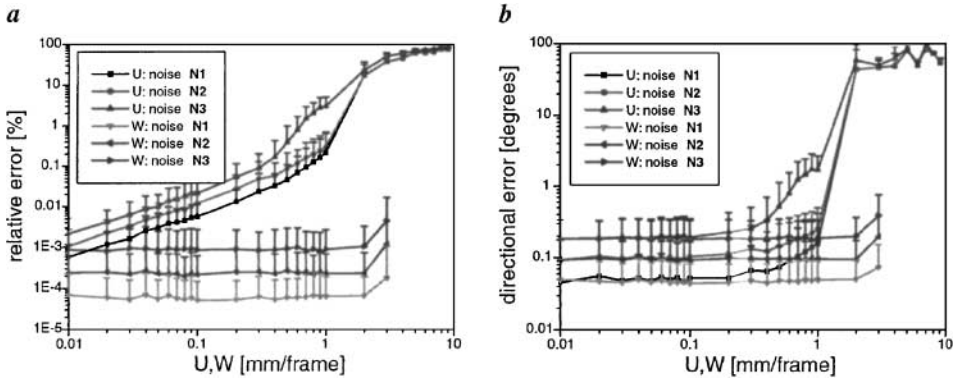

FIG. 11. Error in the regularized range flow on the sphere depending on the velocity in the X- and Z-directions: (a) relative error and (b) directional error.

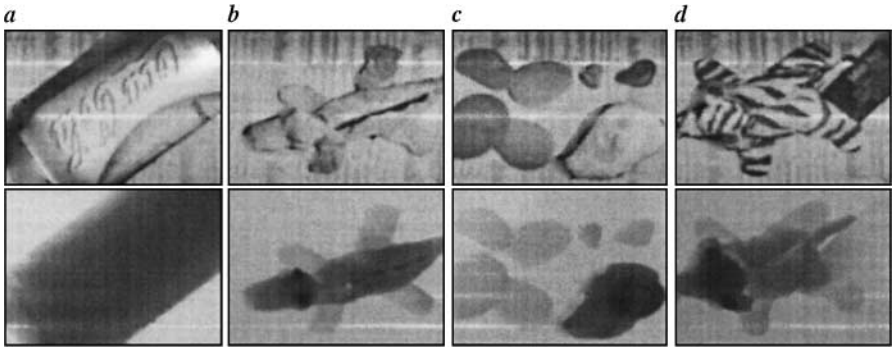

FIG. 12. Real test data using the Biris scanner. Intensity (upper row) and depth data (lower row): (a) a coke can, (b) a toy crocodile, (c) some pebbles, and (d) a toy tiger.

any deformations. The computational accuracy is analyzed for both Biris laser [30] and structured light range data.

For the Biris range sensor we used a number of objects to test the range flow algorithm. The intensity and depth data of a few examples are shown in Fig. 12. On these data sets we obtain the following errors, where the TLS results are shown on the left of the regularized results:

Object	Correct flow [mm]	E_r [%]	E_d [°]	Dens. [%]	E_r [%]	E_d [°]
a	$[0.10\ \ 0.26\ \ 0.22]^T$	2.7 ± 5.5	6.6 ± 4.9	30.4	1.2 ± 0.7	5.3 ± 4.2
b	$[0.13\ \ 0.51\ \ 0.25]^T$	2.2 ± 2.2	4.1 ± 2.9	35.8	1.2 ± 0.8	2.1 ± 1.18
c	$[0.45\ \ -0.25\ \ 0.38]^T$	5.4 ± 7.2	12.0 ± 9.7	28.8	3.8 ± 2.7	10.4 ± 7.2
d	$[0.16\ \ 0.19\ \ -0.32]^T$	0.7 ± 0.9	2.5 ± 2.3	28.8	0.3 ± 0.4	1.1 ± 1.3

Clearly the regularization not only yields 100% dense full flow but also significantly improves upon the results of the initial TLS algorithm. The computed 3D velocity accuracy is quite good.

Next we examine the achievable range flow accuracy using a structured light sensor assembled at the Institute for Botany at the University of Heidelberg [29]. Again some test objects are placed on a set of linear positioning tables and moved by a known displacement between consecutive scans. We examine a sheet of crumpled paper and a freshly cut leaf in more detail. Example intensity and depth data from these two test scenes are given in Fig. 13.

Because it makes a difference whether the motion is in the horizontal or the vertical direction we show the relative and directional errors depending on U, V, and W separately in Fig. 14. We did not observe any difference between movements in X or Y or an influence of the sign. Therefore we plot the results as function of the absolute value of the horizontal or vertical motion. Those movements containing both horizontal and vertical motion are plotted among the horizontal velocities, because we found this component to clearly dominate the error.

For the relative error depending on U or V (Fig. 14a) we find the previously observed relationship with increasing magnitude. Again the regularization does significantly improve the accuracy for small velocities. Of two test objects used, we got better results for the crumpled paper than for the leaf. In order to maintain a relative error below 1% the speed should not exceed 0.5 mm/frame in the horizontal direction. Note that the full flow density

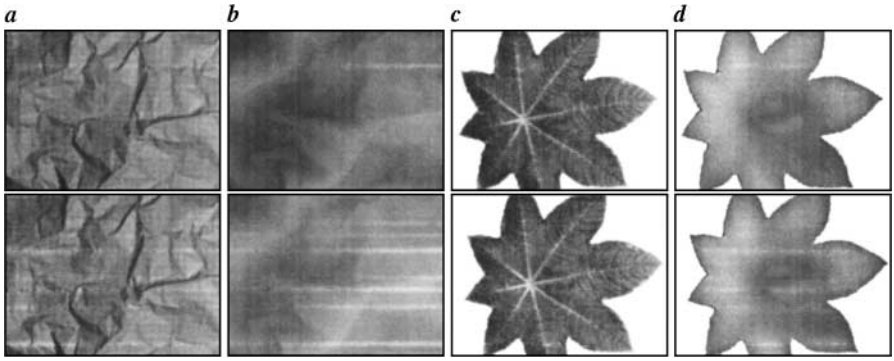

FIG. 13. Real structured light data to assess the range flow performance: the first and fifth frame in an example sequence are shown on top of each other. Crumpled paper $\boldsymbol{f} = [-0.5 \;\; 0.0 \;\; 1.5]^T$ mm: (a) intensity and (b) depth. Freshly cut leaf $\boldsymbol{f} = [1.0 \; 0.0 \; -0.2]^T$ mm: (c) intensity and (d) depth.

on the crumpled paper data is not very high and quickly decreases with increasing velocity. The directional error depending on the horizontal movement (Fig. 14b) shows a similar rise towards higher velocities. To ensure an error below 5° in the estimated velocity direction the speed should not exceed 0.5 mm/frame.

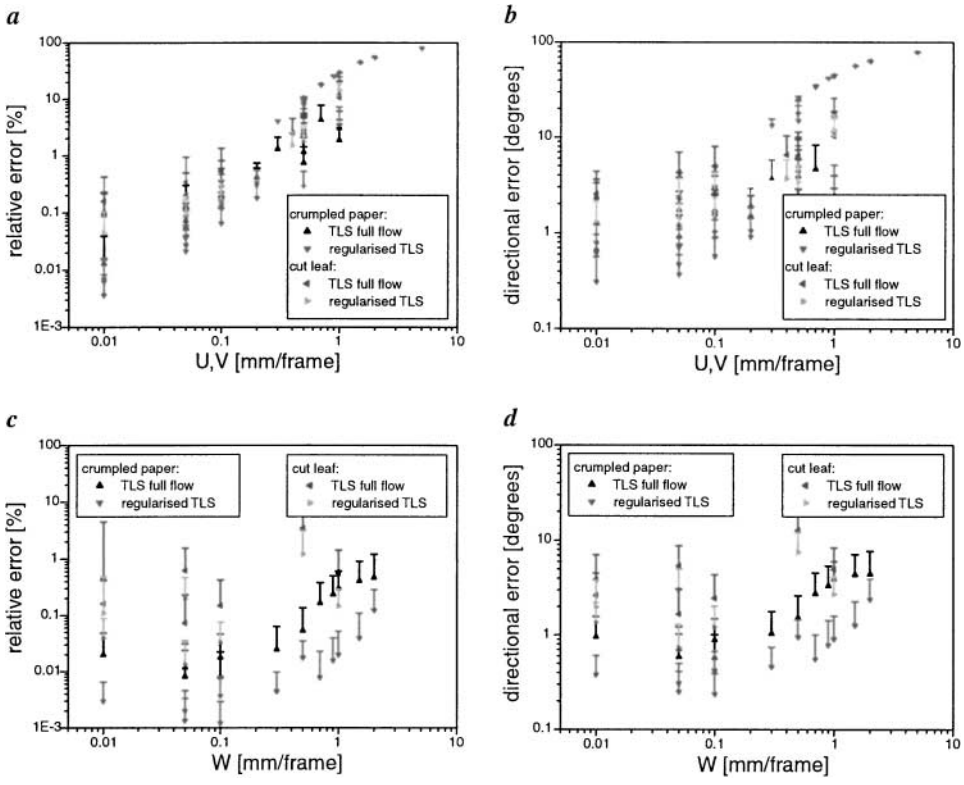

FIG. 14. Results on the real data for different movements: (a) Relative error and (b) directional error depending on the motion in the X- or Y-direction. (c) Relative error and (d) directional error depending in the motion in the Z-direction.

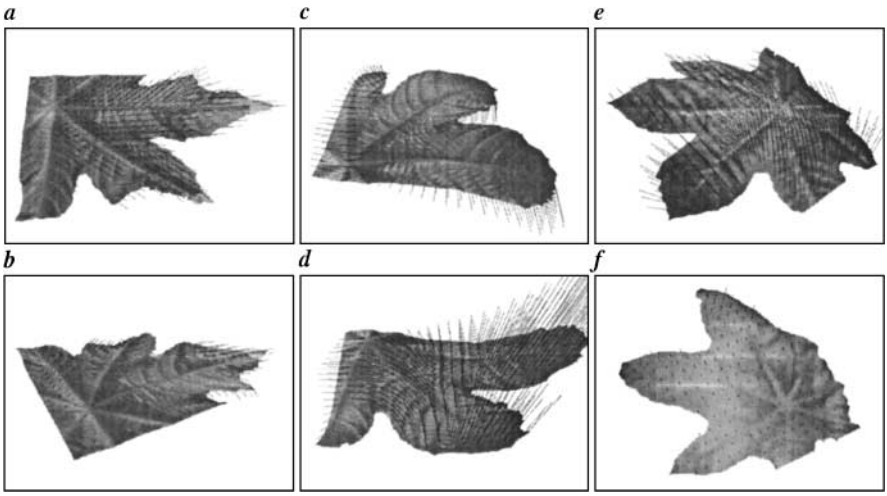

FIG. 15. Example data and movements of castor bean leaves. (a, b, c, d) are taken with the structured light sensor at 3-min sampling and (e, f) with the Biris sensor with a 5-min sampling interval. (a) Fixed leaf viewed from above and (b) the same leaf from underneath. (c) An unconstrained leaf with rapidly folding lobes and (d) the same leaf viewed from underneath. (e) Unconstrained leaf during the night and (f) one at daytime.

For the error in dependence of the vertical velocity we observe only a minor increase for velocities up to 3 mm/frame. This is in qualitative agreement with the results found on synthetic data. However, the errors on real data are significantly higher. Again we see that the regularization improves the results and that we get better results on the crumpled paper than on the leaf. The relative error remains on average well below 1% for the complete velocity range. Except for one outlier we also find the directional error on the regularized flow to be less than 5° for all considered movements.

5.4. Leaf Motion

Finally we show some examples of a scientific application of the proposed range flow estimation scheme. Range flow fields from living castor bean leaves show rather different types of movement under different conditions (Fig. 15). Leaves that are confined in a plane by pulling at the lobe tips are relatively flat but not entirely so (Figs. 15a, 15b). The movement on such a fixed leaf is predominantly in the horizontal plane, but there is some vertical motion as well. In the given example there are locations where the leaf is moving upward and others where it moves downward.

An unconstrained leaf just after nightfall already shows some bending of the lobes (Figs. 15c, 15d). This bending expresses itself in a large downward velocity. Another example of a free leaf taken during the night shows an overall upward movement paired with a fair bit of lateral motion (Fig. 15e). At day time, on the other hand, much smaller velocities are encountered (Fig. 15f).

6. CONCLUSION

An algorithm to compute dense 3D range flow fields from sequences of range data sets has been presented. Towards this end we introduced a constraint equation that can be evaluated

directly on the sensor grid. We showed how a local velocity estimate can be obtained in a total least squares framework. Our technique is able to distinguish areas where line and plane normal as well as full range flow can be computed. It has been shown how the sparse information from the TLS technique can be regularised to yield a dense full flow field. The performance of the proposed algorithm is quantitatively assessed on synthetic and real data and the method has been found to give excellent results. Finally it could be shown that the motion of living castor bean leaves can be nicely captured.

APPENDIX A

Minimum Norm Solutions

We present a way to estimate a parameter vector even when there are linear dependencies in the data. As stated above any vector in the nullspace of $\boldsymbol{J}$ is a possible solution. The only sensible choice is that resulting in an estimate with minimal norm—here we always use the Euclidean norm. Following [25] this estimate can be derived as follows. Assume we have $\lambda_1 > \cdots > \lambda_p > \lambda_{p+1} \approx \cdots \approx \lambda_n \approx 0$; then any linear combination of the eigenvectors $\hat{e}_i; i > p$ is a solution to (17):

$$p = \sum_{i=p+1}^{n} g_i \hat{e}_i = \boldsymbol{E}_p \boldsymbol{g} \quad \text{where } \boldsymbol{E}_p = [\hat{e}_{p+1}, \ldots, \hat{e}_n] = \begin{bmatrix} e_{(p+1)1} & \cdots & e_{n1} \\ \vdots & \ddots & \vdots \\ e_{(p+1)n} & \cdots & e_{nn} \end{bmatrix}. \tag{A.1}$$

The norm of $\boldsymbol{p}$ is then given by

$$\|\boldsymbol{p}\| = \boldsymbol{g}^T \boldsymbol{E}_p^T \boldsymbol{E}_p \boldsymbol{g} = \boldsymbol{g}^T \boldsymbol{g} = \sum_i g_i^2. \tag{A.2}$$

The additional constraint $p_n = 1$ can be expressed as

$$p_n = \left(\sum_{i=p+1}^{n} g_i \hat{e}_i \right)_n = \sum_{i=p+1}^{n} g_i e_{in} = \boldsymbol{v}_n^T \boldsymbol{E}_p \boldsymbol{g} = 1, \tag{A.3}$$

where $\boldsymbol{v}_n = [0, \ldots, 0, 1]^T$. Equations (A.2) and (A.3) can be combined using a Lagrange multiplier to give an energy functional to be minimized:

$$E(\boldsymbol{g}) = \boldsymbol{g}^T \boldsymbol{g} + \lambda \left(\boldsymbol{v}_n^T \boldsymbol{E}_p \boldsymbol{g} \right) \rightarrow \min. \tag{A.4}$$

The minimum is found by setting the partial derivatives of E with respect to the g_i to zero. Doing so yields

$$2g_i + \lambda e_{in} = 0 \rightarrow g_i = -\frac{\lambda}{2} e_{in} \rightarrow \boldsymbol{g} = -\frac{\lambda}{2} \boldsymbol{E}_p^T \boldsymbol{v}_n. \tag{A.5}$$

Substitution into (A.3) gives

$$\lambda = \frac{-2}{\boldsymbol{v}_n^T \boldsymbol{E}_p \boldsymbol{E}_p^T \boldsymbol{v}_n}. \tag{A.6}$$

The minimum norm solution then equates to

$$\boldsymbol{p} = \boldsymbol{E}_p \boldsymbol{g} = \frac{\boldsymbol{E}_p \boldsymbol{E}_p^T \boldsymbol{v}_n}{\boldsymbol{v}_n^T \boldsymbol{E}_p \boldsymbol{E}_p^T \boldsymbol{v}_n} \quad \text{or} \quad p_k = \frac{\sum_{i=p+1}^{n} e_{ik} e_{in}}{\sum_{i=p+1}^{n} e_{in}^2}; \quad k = 1 \ldots n. \tag{A.7}$$

It is sometimes advantageous to express this solution in terms of the eigenvectors corresponding to the nonvanishing eigenvalues. To do so we note that $\boldsymbol{E}_p \boldsymbol{E}_p^T = \mathbb{1} - \bar{\boldsymbol{E}}_p \bar{\boldsymbol{E}}_p^T$, with $\bar{\boldsymbol{E}}_p = [e_1, \ldots, e_p]$. Then (A.7) becomes

$$\boldsymbol{p} = \frac{\left(\mathbb{1} - \bar{\boldsymbol{E}}_p \bar{\boldsymbol{E}}_p^T\right) \boldsymbol{v}_n}{\boldsymbol{v}_n^T \boldsymbol{E}_p \boldsymbol{E}_p^T \boldsymbol{v}_n}. \tag{A.8}$$

In components this equals:

$$p_k = \frac{-\sum_{i=1}^{p} e_{ik} e_{in}}{1 - \sum_{i=1}^{p} e_{in}^2}; \quad k = 1 \ldots n-1 \quad \text{and} \quad p_n = \frac{1 - \sum_{i=1}^{p} e_{in}^2}{1 - \sum_{i=1}^{p} e_{in}^2} = 1. \tag{A.9}$$

The actual parameter vector $\boldsymbol{m} = [p_1 \ldots p_{n-1}]^T$ can also be expressed in a vector equation:

$$\boldsymbol{m} = \frac{\sum_{i=p+1}^{n} e_{in} \left[e_{i1}, \ldots, e_{i(n-1)}\right]^T}{\sum_{i=p+1}^{n} e_{in}^2} = \frac{-\sum_{i=1}^{p} e_{in} \left[e_{i1}, \ldots, e_{i(n-1)}\right]^T}{1 - \sum_{i=1}^{p} e_{in}^2}. \tag{A.10}$$

Thus the parameter solution can be expressed as a linear combination of the "reduced" eigenvectors $\hat{\boldsymbol{b}}_k$. By reduced we mean those vectors obtained from the first $n-1$ elements of the eigenvectors: $\hat{\boldsymbol{b}}_k = \frac{1}{\sum_{i=1}^{n-1} e_{ki}^2} [e_{k1} \ldots e_{k(n-1)}]^T$. In case only the smallest eigenvalue vanishes; i.e., all parameters can be estimated. (A.10) becomes ($p = n-1$)

$$\boldsymbol{m} = \frac{e_{nn} \left[e_{n1}, \ldots, e_{n(n-1)}\right]^T}{e_{nn}^2} = \frac{1}{e_{nn}} \begin{bmatrix} e_{n1} \\ \vdots \\ e_{n(n-1)} \end{bmatrix}. \tag{A.11}$$

This is consistent with the previously given solution (23).

ACKNOWLEDGMENT

This work has been funded under the DFG research unit "Image Sequence Analysis to Investigate Dynamic Processes" (FOR240) and an individual NSERC (National Science and Engineering Research Council of Canada) research grant.

REFERENCES

1. S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, Three-dimensional scene flow, in *ICCV, Pittsbrugh, PA, September 2000*, pp. 722–729.
2. Y. Zhang and C. Kambhamettu, Integrated 3d scene flow and structure recovery from multiview image sequences, in *CVPR, 2000*, Vol. 2.
3. R. Carceroni and K. Kutulakos, Multi-view scene capture by surfel sampling: From video streams to non-rigid 3d motion, shape & reflectance, in *ICCV, Vancouver, Canada, July 2001*, pp. 60–67.

4. M. Harville, A. Rahimi, T. Darrell, G. Gordon, and J. Woodfill, 3d pose tracking with linear depth and brightness constraints, in *ICCV, 1999*, pp. 206–213.
5. B. K. P. Horn and J. Harris, Rigid body motion from range image sequences, *CVGIP* **53**(1), 1991, 1–13.
6. Y. Liu and M. A. Rodrigues, Correspondenceless motion estimation from range images, in *ICCV, 1999*.
7. L. Lucchese, G. Doretto, and G. M. Cortelazzo, Frequency domain estimation of 3-d rigid motion based on range and intensity data, in *Recent Advances in 3-D Digital Imaging and Modeling, Ottawa. Ontario, 1997*, pp. 107–112.
8. B. Sabata and J. K. Aggarwal, Estimation of motion from a pair of range images: A review, *CVGIP* **54**(3), 1991, 309–324.
9. R. Szeliski, Estimating motion from sparse range data without correspondence, in *ICCV, 1988*, pp. 207–216.
10. L. Tsap, D. Goldgof, and S. Sarkar, Model-based force-driven nonrigid motion recovery from sequences of range images without point correspondences, *Image Vision Comput.* **17**(14), 1999, 997–1007.
11. L. Tsap, D. Goldgof, and S. Sarkar, Multiscale combination of physically-based registration and deformation modeling, in *CVPR, Hilton Head, SC, June 2000*, Vol. 2, pp. 422–429.
12. M. Yamamoto, P. Boulanger, J. Beraldin, and M. Rioux, Direct estimation of range flow on deformable shape from a video rate range camera, *Pattern Anal. Math. Intell.* **15**(1), 1993, 82–89.
13. J. L. Barron, D. J. Fleet, and S. Beauchemin, Performance of optical flow techniques, *Int. J. Comput. Vision* **12**(1), 1994, 43–77.
14. H. Haußecker and H. Spies, Motion, in *Handbook of Computer Vision and Applications* (B. Jähne, H. Haußecker, and P. Geißler, eds.), Vol. 2, Chap. 13, Academic Press, San Diego, 1999.
15. B. K. P. Horn and B. Schunk, Determining optical flow, *Artif. Intell.* **17**, 1981, 185–204.
16. K. Chaudhury, R. Mehrota, and C. Srinivasan, Detecting 3d flow, in *Conference on Robotics and Automation, May 1994*, Vol. 2, pp. 1073–1078.
17. R. Klette, A. Koschan, and K. Schlüns, *Computer Vision Räumliche Information aus digitalen Bildern*, Vieweg, Braunschweig, 1996.
18. B. Jähne, *Digital Image Processing*, 3rd ed., Springer-Verlag, Berlin, 1995.
19. J. L. Barron, A. Liptay, and H. Spies, Optical and range flow to measure 3d plant growth and motion, in *Image Vision Computing New Zealand, Hamilton, New Zealand, November 2000*, pp. 68–77.
20. J. L. Barron and H. Spies, Quantitative regularized range flow, in *Vision Interface, Montreal, Canada, May 2000*, pp. 203–210.
21. H. Spies, B. Jähne, and J. L. Barron, Dense range flow from depth and intensity data, in *ICPR*, *Barcelona, Spain, September 2000*, pp. 131–134.
22. B. Jähne, H. Haußecker, H. Scharr, H. Spies, D. Schmundt, and U. Schurr, Study of dynamical processes with tensor-based spatiotemporal image processing techniques, in *ECCV*, pp. 322–336, Springer-Verlag, 1998.
23. H. Haußecker, C. Garbe, H. Spies, and B. Jähne, A total least squares for low-level analysis of dynamic scenes and processes, in *DAGM, Bonn, Germany, 1999*, pp. 240–249, Springer-Verlag.
24. H. Spies, B. Jähne, and J. L. Barron, Regularised range flow, In editor, *ECCV, Dublin, Ireland, June, July 2000,* (D. Vernon, Ed.), Lecture Notes in Computer Science, Vol. 1843, Part 2, pp. 785–799, Springer-Verlag.
25. S. Van Huffel and J. Vandewalle, *The Total Least Squares Problem: Computational Aspects and Analysis*, Soc. Industrial and Appl. Math., Philadelphia, 1991.
26. W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C*, 2nd ed., Cambridge Univ. Press, Cambridge, MA, 1992.
27. H. Spies, H. Haußecker, B. Jähne, and J. L. Barron, Differential range flow estimation, in *DAGM, Bonn, Germany, September 1999*, pp. 309–316.
28. J. Beraldin, S. F. El-Hakim, and F. Blais, Performance evaluation of three active vision systems built at the National Research Council of Canada, in *Conf. on Optical 3D Measurement Techniques III, Vienna, Austria, October 1995*, pp. 352–361.
29. H. Spies, *Analysing Dynamic Processes in Range Data Sequences*, Ph.D. thesis, University of Heidelberg, Heidelberg, July 2001.
30. M. Rioux and F. Blais, Compact three-dimensional camera for robotic applications, *J. Opt. Soc. Amer. A* **3**(9), 1986, 1518–1521.