Error Analysis of Robust Optical Flow Estimation by Least Median of Squares Methods for the Varying Illumination Model

Yeon-Ho Kim and Avinash C. Kak

Abstract—The apparent pixel motion in an image sequence, called optical flow, is a useful primitive for automatic scene analysis and various other applications of computer vision. In general, however, the optical flow estimation suffers from two significant problems: the problem of illumination that varies with time and the problem of motion discontinuities induced by objects moving with respect to either other objects or with respect to the background. Various integrated approaches for solving these two problems simultaneously have been proposed. Of these, those that are based on the LMedS (Least Median of Squares) appear to be the most robust. The goal of this paper is to carry out an error analysis of two different LMedS-based approaches, one based on the standard LMedS regression and the other using a modification thereof as proposed by us recently. While it is to be expected that the estimation accuracy of any approach would decrease with increasing levels of noise, for LMedS-like methods, it is not always clear as to how much of that decrease in performance can be attributed to the fact that only a small number of randomly selected samples is used for forming temporary solutions. To answer this question, our study here includes a baseline implementation in which all of the image data is used for forming motion estimates. We then compare the estimation errors of the two LMedS-based methods with the baseline implementation. Our error analysis demonstrates that, for the case of Gaussian noise, our modified LMedS approach yields better estimates at moderate levels of noise, but is outperformed by the standard LMedS method as the level of noise increases. For the case of salt-and-pepper noise, the modified LMedS method consistently performs better than the standard LMedS method.

Index Terms—Optical flow, robust estimation, varying illumination, least median of squares method, error analysis.

1 Introduction

The estimation of motion in a sequence of images is a basic task in computer vision with many interesting applications, such as motion segmentation, extracting structure from motion, video surveillance, image compression, robot navigation, etc. Despite the increasing interest in motion features, accurate motion estimation still remains challenging primarily because intensity variations at a given pixel can also be caused by temporal changes in illumination, besides, of course, the relative motion between the camera and the scene. Another difficulty is created by the fact that, if an object surface is lacking in any visual detail, there will be no measurable apparent motion between the camera and the scene at the pixels corresponding to such surfaces. The dependence of the accuracy of an optical flow measurement on the local object surface detail and the needed spatial extent of that detail is referred to as the *aperture problem*. This problem was first stated by Stumpf in [1] and has been generalized in different ways by others [2], [3], [4]. Unfortunately, the larger the aperture, the more likely that there exist unrelated or erroneous motions that would corrupt the correct estimation of the motion in a given aperture.

There exist simple solutions to the problem of optical flow estimation if one can ignore frame-to-frame variations in illumination. These solutions, usually referred to as the differential optical flow estimation techniques, calculate optical flow by minimizing the error of a constraint equation that sets the time-derivative of image intensity to zero. Such solutions are represented by the contributions of Horn and Schunck [3] and Lucas and Kanade [5]. As these methods are based on the assumption of constant illumination, they perform poorly when illumination varies from frame to frame since now we cannot enforce the $(dI/dt = 0)$ constraint. Since the work of Horn and Schunck in [3] and Lucas and Kanade in [5], there have been other research contributions that have taken into account frame-to-frame variations in illumination [6], [7], [8], [4].

All algorithms for optical flow calculation—whether or not they take illumination variations into account—are also based on the assumption that optical flow is locally smooth. This is frequently referred to as the motion smoothness assumption. In the formulation of optical flow, as originally presented by Horn and Schunck, this assumption is tantamount to saying that the optical flow field is continuous [3]. In general, this will be the case when the scene consists of only one convex object (whose surfaces are textured in some sense) and the relative motion between the camera and the object is smooth. While the enforcement of this assumption mitigates somewhat the aperture problem, the assumption often cannot be satisfied by real-world imagery. If included in an aperture are two objects executing different motions, the smoothness assumption could easily be violated and the estimated optical flow may be wrong, especially at a motion boundary. Many methods based on the concept of robust estimation have been proposed to cope with this multiple motion problem [9], [2], [10].

• *The authors are with the Robot Vision Laboratory, School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907. E-mail: {yeonho, kak}@ecn.purdue.edu.*

Manuscript received 26 Apr. 2005; revised 1 Dec. 2005; accepted 4 Jan. 2006; published online 13 July 2006.
Recommended for acceptance by M. Srinivasan.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0215-0405.

For obvious reasons, it would be desirable to solve these two problems simultaneously—the problem caused by large illumination variations and the problem caused by large motion discontinuities—and there have been some studies in the past to fulfill this need [11], [12], [13], [14], [15], [16]. The proposed integrated approaches combine an illumination model with a robust estimation technique. These approaches can be classified into the following two groups.

One group consists of the global approaches that estimate optical flow robustly by minimizing a regularization function for all the pixels in an image. For instance, the methods in [2], [11], [13] use an M-estimator instead of the nonrobust quadratic estimator used previously by Horn and Schunck [3].

The other group of robust estimation techniques is based on the local approaches that employ robust estimation to solve a set of overdetermined linear equations for pixels in a local area of the image. Examples are the methods reported in [9], [12], [10]. These algorithms use the least median of squares (LMedS) method instead of the (more classical) least sum of squares (LS) method to solve a regression problem.

All the LMedS-based optical flow estimation methods cited above, including the one proposed by us [12], use a small number n of randomly selected pixels in a local area of the image to form a temporary solution for the motion estimates. Subsequently, for a given local window, one of the s such temporary solutions is selected on the basis of what residual error it yields vis-a-vis all the other pixels in the local window. (Note that each temporary solution comes from one sampling taken from the pixels in the local window.) Both the numbers, n and s, are kept small to reduce the overall computational cost. Although the LMedS methods have the desired asymptotic properties with respect to both n and s—the estimation error decreases monotonically as both increase—the accuracy achieved in any given implementation of LMedS would obviously depend on actual values used for n and s. It is this dependence of the estimation accuracy on n and s that is the focus of our error analysis. We also want to understand how this dependency changes as the amount of noise in an image sequence changes and also as the type of that noise changes.

For the sake of completeness, we must mention a prior error analysis report by Ong and Spann [10] where they used Gaussian-noise corrupted image sequences to analyze the accuracy of the standard LMedS method for motion estimation. This study, however, did not investigate the dependence of the estimation accuracy on the number of pixels used for the formation of temporary solution and the number of samplings in a local window.

Since the primary focus of our error-analysis is the dependence of the motion estimation errors on the number of pixels used for forming temporary solutions and on the number of samplings used in a local window, we obviously need a baseline approach that constructs a motion estimate on the basis of *all of the information in a local window*. The estimates achieved with any LMedS method could then be compared with the results obtained with the baseline method for different levels of noise. The baseline implementation presented in this paper uses all the pixels in the local window to construct a set of all the possible temporary solutions. A temporary solution for this implementation is obtained using the LS method on a sufficient number of overdetermined constraint equations in a subwindow in the local window. However, using all the pixels in the baseline method increases its computational cost excessively. We therefore employ a fast median calculation technique based on Huang et al.'s algorithm [17]. This algorithm uses the notion of a "running" window-to-window calculation of the median. The basic idea here is to save computation time by reusing the previous calculation(s) for overlapped data when a local window moves to the next position.

We then compare the estimation errors of the two LMedS methods—one based on the standard LMedS regression and the other using a modification thereof as proposed by us recently—with the baseline implementation. Our error analysis demonstrates that, for the case of Gaussian noise, our modified LMedS approach yields better estimates at moderate levels of noise, but is outperformed by the standard LMedS method as the level of noise increases. For the case of salt-and-pepper noise, the modified LMedS method consistently performs better than the standard LMedS method. This conclusion is based on the error analysis using a real image sequence for which the ground truth motion is given. Two different types of image noise are used for this error analysis. One is the detector noise that is always present to varying degrees in recorded images; this type of noise is generally modeled by a zero mean Gaussian process. The other type of noise we incorporate in the images is data drop-out noise (commonly referred to as the salt-and-pepper noise) that is caused by errors in data transmission.

The rest of this paper is organized as follows: In Section 2, we review separately the previously proposed robust techniques to estimate optical flow in the presence of illumination changes and in the presence of multiple motions. In Section 3, we then briefly present previously published integrated approaches; this review also mentions our global and local integrated methods for dealing with illumination changes and with motion boundaries. In Section 4, for the aforementioned baseline implementation, we present a nonstatistical LMedS technique that uses a modified version of the previously proposed LMedS method and a running calculation of the median. Section 5 presents error analysis of the LMedS methods by comparing the estimation errors for the synthetic and real image sequences contaminated by Gaussian and salt-and-pepper noise. Section 6 concludes the paper.

2 Robust Optical Flow Estimation under Varying Illumination

Following Horn and Schnuck's method in [3], most differential optical flow estimation methods are based on the analytical formulation of the spatial and temporal derivatives of the image intensity under the brightness constancy assumption. The brightness constancy assumption can be stated as the standard optical flow constraint equation $I_x u + I_y v + I_t = 0$ where I is the image intensity (which may be the output obtained after some low-pass or band-pass filtering) at a point (x, y) in the image at time t, and where (u, v), with $u = \lim_{\delta t \to 0} \frac{\delta x}{\delta t}$ and $v = \lim_{\delta t \to 0} \frac{\delta y}{\delta t}$, is the optical flow vector that needs to be estimated. In and of itself, this constraint equation is insufficient for the estimation of the two variables u and v.

The gradient-based methods that have been proposed over the years to solve for u and v in the optical flow constraint equation can be distinguished on the basis of the additional constraints the authors have used for tackling the aperture problem: 1) *global approaches* that use all the pixels in the image and 2) *local approaches* that only use the local information in the vicinity of where optical flow is to be estimated. The global methods proposed in [2], [6], [3], [18] assume that the motion field varies smoothly over the entire image. This assumption then translates into the addition of a global smoothness constraint to the constant brightness equation. The parameters u and v can then be optimally estimated by a regularization that uses both the constraint equations—the original optical flow constraint equation and the global smoothness constraint equation.

Global smoothing-based approaches do not work well with real-world imagery mainly because they assume that exactly the same degree of motion smoothing is needed everywhere. In order to yield a stable solution, the global approaches must incorporate sufficient smoothing to deal with the worst-case motion discontinuities in an image sequence by adjusting the parameters that define the weight of the smoothness factor. Unfortunately, there exists no clear analytical formulation to decide the value of smoothness parameters and they must be tuned experimentally depending on the spatial properties of image intensities.

For that reason, methods that use only local constraints on motion are better suited to real images. Such methods usually try to fit estimated optical flows to 2D parametric models of motion [5], [18], [19]. While local methods using parametric motion models possess the twin advantages of relatively low computational cost (which is due to their simplicity) and immunity to the problems caused by global smoothing, they still require that the overdetermined equations be independent. If the equations are not sufficiently independent, the solution cannot be found reliably. Due to this reason, the optical flow obtained by local approaches is often sparse. This is a major shortcoming of the local methods.

The global and the local methods cited above are based on the constant brightness assumption. To the extent this assumption is violated, these methods fail to estimate the correct optical flow field. When illumination is not constant, spatially or temporally, the constant brightness assumption will obviously be violated. Varying illumination will obviously cause the image intensity at a pixel to change during motion. When this happens, what we have is $I_x u + I_y v + I_t \neq 0$. This has prompted researchers to propose other constraints that are derived from assumed (and simplified) models of illumination change [20], [6], [21], [22] for solving for u and v. For example, Gennert and Negahdaripour have modeled temporal illumination changes as a combination of multiplicative and additive effects [6]. This they have accomplished by introducing a new optical flow constraint equation: $I_x u + I_y v + I_t = mI + c$, where m is for the multiplicative illumination effects and c for the additive ones. Using this new constraint equation, they estimate the optical flow iteratively using the basic differential framework through the minimization of

$$E = \iint (\nabla I^T \boldsymbol{u} + I_t)^2 + \lambda_s(\|\nabla u\| + \|\nabla v\|) + \lambda_m(\|\nabla m\|) + \lambda_c(\|\nabla c\|)\, dx\, dy,$$

where $\|\nabla u\|$ is an L2-norm of the gradient of u and is defined by $\|\nabla u\| \equiv u_x^2 + u_y^2$, where $u_x = \lim_{\delta x \to 0} \frac{\delta u}{\delta x}$, and other norms are defined in the same way. In [23], Negahdaripour and Yu obtain a set of linear equations by assuming that the values of u, v, m, and c are constant within small regions around each point. The resulting ill-conditioned set of equations is solved by using a least squares method.

With regard to the presence of multiple motions, the problems caused by such motions obviously cannot be dealt with by the local methods that use simplified parametric motion models consisting of single motions—which is often case in the publications that have used such models. When multiple motions are present, it is not possible to assume that the motion is globally smooth since there will exist motion discontinuities at the boundaries between the different objects. Motion estimation in such cases is best carried out with robust estimators instead of those based on the least-squares methods employed by, say, Horn and Schunck in [3] and by Lucas and Kanade in [5]. The robustness of an estimator—meaning the sensitivity to outliers—can be quantified by the breakdown point that is the fraction of outliers that can corrupt the estimator [24]. For example, the breakdown point of least-squares methods—also referred to as quadratic estimators in discussions related to robustness —is zero because a single outlying data point can completely mislead the estimator. Because of this sensitivity of the least-squares-based estimators, one must take recourse to robust techniques when multiple motions are present.

The optical flow estimation methods using the robust estimation techniques can again be grouped as global approaches and local approaches—this is the same characterization we used earlier for the least squares-based methods. An example of a robust approach that is global is the work of Black and Anandan [2]; they have shown how a maximum likelihood estimator (M-estimator) can be used for this purpose. However, the breakdown point of the M-estimator is relatively low, $1/(p+1)$, where p is the number of parameters to be estimated [24]. An additional issue related to the M-estimator is that it does not entirely eliminate the global smoothing effect associated with the global implementation of the least-squares approaches since the optical flow in the region where the gradient of the intensity is high "bleeds" into the region where the gradient of the intensity is low, although the extent of this bleeding depends on the weight of the smoothness term. Yet, another shortcoming of the Black and Anandan method is that it has many parameters that need to be tuned carefully; a challenging task since there does not exist a clearly defined relationship between the value of the parameters and the performance of the algorithm.

In the category of the robust approaches that are local, the most commonly used are the iterative reweighted least squares (IRLS) method described in [25], [14], [15], [26], the voting-based method described in [27], [28], and the least median of squares (LMedS) method described in [9], [10], [12].

The LMedS implementations are based on the work of Rousseeuw and Leroy in [24]; these possess a breakdown point of 50 percent which is at the upper limit in robust statistics. It needs to be stated that the LMedS-based method is superior to the IRLS method (IRLS requires a good initial guess) and the voting-based method (which tends to be computationally burdensome). LMedS-based estimation of optical flow has been carried out by Ong and Spann [10] and Bab-Hadiashar and Suter [9]. In their implementations, a temporary solution that minimizes the median of the residual errors is first sought from a set of randomly sampled pairs of pixels in a local area. Then, a final solution is obtained using the weighted least-squares method with the weights that are calculated from the temporary solution. Though these LMedS-based implementations are very robust and free from a need for parameter tuning, they are slowed by the need to estimate the medians.

3 Integrated Methods—Global versus Local

Whereas the various methods reviewed in the previous section seek to estimate the optical flow in the presence of large illumination changes and when there exist motion discontinuities, they do not solve the two problems simultaneously. In other words, those methods cannot be used when an image sequence, recorded under varying frame-to-frame illumination, contains motion discontinuities. If we use the work reported in [6] and [4] to model the radiometric dependence of the motion transformation of an object under varying illumination, the assumptions embedded in the constraint equations (i.e., the "relaxed" brightness constant constraint and the smoothness constraints) would nevertheless be violated at motion discontinuities in a sequence of images. On the other hand, the robust optical flow estimation techniques of [2], [10], while able to cope with such motion discontinuities, would result in inaccurate estimation of optical flow in the presence of frame-to-frame illumination changes.

For obvious reasons, it would be desirable to solve these two problems simultaneously. To fulfil this need, some methods have indeed been proposed [11], [14], [15], [16]. But, some of the suggested integrated approaches are limited to specialized tasks such as image registration as in [14] or change recovery as in [11], whereas others use the simplifying assumption that illumination changes by either a multiplicative or an additive factor, but not both [15], [16]. To remedy this problem, we previously proposed two different methods: A global integrated method in [13] and a local integrated method in [12]. These combine a more complete varying illumination model (in the sense that it deals with both multiplicative and additive factor of illumination changes) and the robust estimation framework that is designed for the general motion estimation. In this section, we present a more detailed comparison of our two methods.

3.1 A Global Integrated Method

To estimate optical flow robustly in the presence of both illumination variations and motion discontinuities, a previous contribution from us in [13] integrated the Gennert and Negahdaripour's varying illumination model in [6] with Black and Anandan's robust estimation framework in [2]. This approach is global as it seeks to minimize the following regularization function over all the pixels of the image

$$E = \iint \left\{ \begin{array}{c} \rho(I_x u + I_y v + I_t - mI - c, \sigma_d) \\ +\lambda_s(\rho(\nabla u, \sigma_s) + \rho(\nabla v, \sigma_s)) \\ +\lambda_m \rho(\nabla m, \sigma_m) + \lambda_c \rho(\nabla c, \sigma_c) \end{array} \right\} dx\, dy, \quad (1)$$

where $I_x u + I_y v + I_t - mI - c$ is the error term for the brightness constraint used, I_x, I_y, and I_t the spatial and temporal partial derivatives of the image, u and v the motion parameters (i.e., optical flow), and m and c the radiometric parameters for the multiplicative and additive terms for modeling the varying illumination. This function essentially tries to regularize the brightness constraint with three smoothness constraints. The smoothness constraints are with respect to the gradient of the parameters u, v, m, and c. The relative weight of each smoothness constraint is controlled by the respective λ. The function $\rho()$ applied separately to each of the constraints is meant to deal with the outliers at the motion boundaries. We use the Lorentzian function for this purpose; it is defined as $\rho(x, \sigma) = \log(1 + \frac{1}{2}(\frac{x}{\sigma})^2)$, where σ controls the weight to be given to the variable x. Using the same technique as in [2], we can find the global minimum of the objective function E by first choosing a large value for σ, which creates a convex approximation to E and then decreasing the σ values in order to find a more accurate minimum. This process is repeated until we converge to a solution.

As we demonstrated in [13], the above integrated minimization yields results superior to what can be obtained by separately applying the two minimizations needed (one for dealing with motion discontinuities and the other for illumination variations). Nonetheless, we are still faced with the cumbersome task of a trial-and-error determination of the λs—an issue that is an unpleasant but inherent drawback of any global method. More recently, we have therefore focused our efforts on designing local methods that we present next.

3.2 Local Integrated Methods

As we reviewed in Section 2, the most commonly used method for the local optical flow estimation uses the least sum of squares (LS) method. If we denote the parameters u, v, m, and c in the form of a vector $\boldsymbol{\theta} = [u\ v\ m\ c]^T$, then the LS solution can be written as below:

$$\hat{\boldsymbol{\theta}}(x, y) = \arg\min_{\boldsymbol{\theta}} \sum_{(x_i, y_i) \in R(x,y)} r^2(x_i, y_i, \boldsymbol{\theta}), \quad (2)$$

where $R(x, y)$ is the local region centered at (x, y), i an index for the pixels in the region, and $r(x, y, \boldsymbol{\theta})$ the residual error for the pixel at (x, y) as defined by the following equation:

$$\begin{aligned} r(x, y, \boldsymbol{\theta}) &= I_x(x, y) \cdot u + I_y(x, y) \\ &\quad \cdot v + I_t(x, y) - I(x, y) \cdot m - c & (3) \\ &= [I_x(x, y)\ I_y(x, y)\ \ -I(x, y)\ \ -1] \\ &\quad \cdot [u\ v\ m\ c]^T + I_t(x, y) & (4) \\ &= \boldsymbol{a}^T(x, y) \cdot \boldsymbol{\theta} + b(x, y). & (5) \end{aligned}$$

Substituting (3) in (2), we get

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{(x_i, y_i) \in R} (\boldsymbol{a}^T(x_i, y_i) \cdot \boldsymbol{\theta} + b(x_i, y_i))^2. \quad (6)$$

A pseudoinverse solution to the above minimization is given by $\hat{\boldsymbol{\theta}} = (\boldsymbol{A}_R^T \cdot \boldsymbol{A}_R)^{-1} \cdot \boldsymbol{A}_R^T \cdot (-\boldsymbol{B}_R)$, where

$$\begin{aligned} \boldsymbol{A}_R &= \left[\boldsymbol{a}_1^T\ \boldsymbol{a}_2^T\ \ldots\ \boldsymbol{a}_i^T\ \ldots\ \boldsymbol{a}_N^T\right]^T \text{ and} \\ \boldsymbol{B}_R &= [b_1\ b_2\ \ldots\ b_i\ \ldots\ b_N]^T. \end{aligned} \quad (7)$$

In (7), $\boldsymbol{a}_i^T = \boldsymbol{a}^T(x_i, y_i) = [I_x(x_i, y_i)\ I_y(x_i, y_i)\ \ -I(x_i, y_i)\ \ -1]$ and $b_i = b(x_i, y_i) = I_t(x_i, y_i)$ and N is the total number of pixels in R.

In [2] and [9], Black and Anandan and Bab-Hadiashar and Suter represent the brightness constraint, $I_x u + I_y v + I_t = 0$,

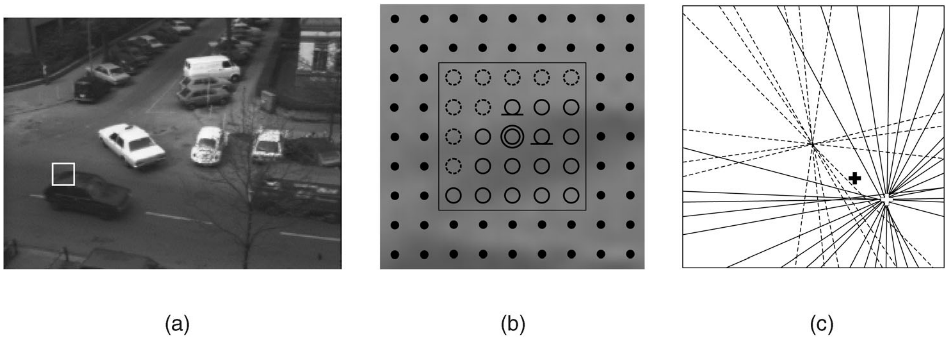


Fig. 1. (a) A frame of an image sequence, (b) pixels in a local window, and (c) constraint lines in the parameter space.

for the motion parameters u and v at each pixel with a line in the (u, v) parameter space and visualize the LS solution as a point with respect to those lines. Although they deal with only the motion parameters, that is, they do not include the radiometric parameters, their representations are good to show the weakness of the LS method to the outliers in the presence of multiple motions. We will now use this representation to provide the reader with a basic insight into the difference between the LS type solution as given by (7) above and the LMedS solution that we will present shortly.

Fig. 1a is a frame of an image sequence taken of a scene in which three vehicles are in motion at the same time. Image pixels in a small region of arbitrary size, outlined with a white square in Fig. 1a, are shown with dots and circles in Fig. 1b. The goal is to estimate the motion parameters at the center of this region. Toward that end, we define a local window, whose size can be varied under program control but is set typically to 5×5, centered in the outlined region, as shown in the figure. Pixels inside this local window are shown with the help of small circles and the pixels outside as black dots. The central pixel where motion estimation will be carried out is shown double-circled.

When a local window straddles the boundary that a moving object forms with either the background or with other moving objects, the optical flow at each pixel in the local window will correspond either to the foreground moving object or to the background, which may either be stationary or another moving object. In the depiction in Fig. 1b, the pixels that are shown with solid circles follow the motion of the moving car and the pixels shown with dashed circles correspond to the stationary background. Obviously, the double-circled pixel at the center of the local window is also following the motion of the car.

As described in [2], [9], the optical flow constraint at each pixel in the local window can be represented by a straight line in the (u, v) parameter space, as illustrated in Fig. 1c. These straight lines correspond to (3) with r set to 0.[1] The dotted lines in Fig. 1c correspond to the constraints at the pixels belonging to the background and the solid lines to the pixels belonging to the moving car. The motion that satisfies two constraint equations at two different pixels in the local window can be represented as the intersection of the two constraint lines in the parameter space. Therefore, as shown in Fig. 1c, the motion of the moving car is given by the intersections of all the solid lines and the motion of the background by the intersection of all the dotted lines in the parameter space.

Now, using Fig. 1c, let's picture what the LS method accomplishes with respect to the two correct solution points shown there. The error minimization that is carried out by the LS method makes no distinction between the errors incurred in satisfying the constraint equations at the solid circles, on the one hand, and at the dotted circles, on the other. In other words, the LS approach treats all the pixels in the local window of Fig. 1b in exactly the same manner. This causes the LS approach to yield a single solution that is some sort of a compromise between the two correct solutions shown in Fig. 1c. A possibility for the constraint lines depicted in Fig. 1c could be the point marked with a black cross. On the other hand, a robust motion estimation approach, as we will see in the next section, will discriminate between the different possible solutions in the parameter space and will return an answer that is close to the dominant solution. For the depiction in Fig. 1c, the dominant solution would correspond to the white cross.

3.2.1 The Standard LMedS Method

In this section, we will first briefly review the robust alternatives to the least-squares-based local methods. In the next section, we will mention the approach we presented in [12] for an integrated local solution to the problem of multiple motion estimation under varying illumination.

What may be referred to as the standard LMedS method for estimating optical flows was proposed by Bab-Hadiashar and Suter and by Ong and Spann [9], [10]. Their work was based on the algorithm proposed by Rousseeuw and Leroy in [24] for a robust solution to the standard regression problem in statistics. The method described in [9], [10] can be described as follows.

The first step of the method consists of choosing p number of pixels randomly from the local window surrounding the pixel where a motion estimate needs to be carried out. The value of p is the number of parameters to be estimated. This number would be four for the local integrated approach in which the parameters to be estimated at each pixel are u, v, m, and c. A temporary solution is calculated from the p number of linear independent equations for the chosen pixels. The next step consists of calculating the residual error at each pixel in the local window using the temporary solution for the parameters. One then finds the median of the square of the error values. This random sampling of the pixels and the

1. Equation (3) will give rise to hyperplanes in the four-dimensional space formed by the parameters u, v, m, and c. The straight lines shown in Fig. 1c are with respect to just the parameters u and v. This is for convenience in visualization. Actual calculations take place in the four-dimensional space spanned by u, v, m, and c.

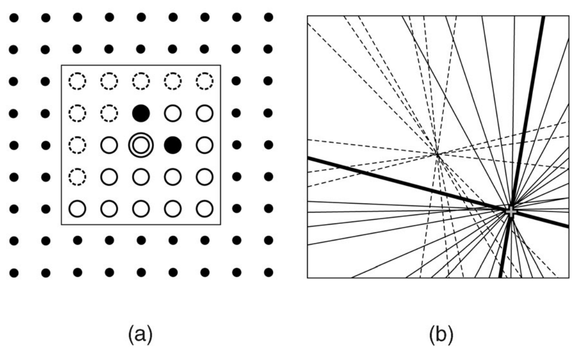


Fig. 2. (a) A local window and (b) constraint lines in the parameter space used for the standard LMedS method.

median calculation is repeated several times. Then, the set of parameters that corresponds to the minimum of the medians is chosen as the final LMedS solution.

The algorithm as described above would be computationally much too expensive if implemented directly. Therefore, in what follows, we will first describe the method in greater detail. This will also allow us to point out how the algorithm can be implemented efficiently and to also introduce the notation we need to present our own work later in this section.

The standard LMedS method in [9], [10] consists of the following minimization:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\tilde{\boldsymbol{\theta}}_k} \left\{ \underset{(x_i,y_i)\in R}{median}\; r^2(x_i, y_i, \tilde{\boldsymbol{\theta}}_k) | k = 1, 2, \ldots, s \right\}, \qquad (8)$$

where $k = \{1, 2, \ldots, s\}$ indexes each different p-pixel sampling from the local window. The local window in this minimization is denoted by R and $r(x, y, \tilde{\boldsymbol{\theta}}_k)$ is the residual error for the pixel at (x, y) with the temporary solution $\tilde{\boldsymbol{\theta}}_k$ as defined in (3) in the previous section. The temporary solution is obtained directly from the p number of linear independent equations corresponding to the p number of pixels at the kth sampling as the following equation: $\tilde{\boldsymbol{\theta}}_k = \boldsymbol{A}_{S_k}^{-1} \cdot (-\boldsymbol{B}_{S_k})$, where, in accordance with (7), S_k is the set of p pixels for the kth sampling and $\boldsymbol{A}_{S_k}$ the $p \times p$ matrix for the S_k. For the local integrated method, $\boldsymbol{A}_{S_k}$ and $\boldsymbol{B}_{S_k}$ can be defined as follows for the estimation of motion and radiometric parameters u, v, m, and c ($p = 4$):

$$\boldsymbol{A}_{S_k} = \left[\boldsymbol{a}_{k,1}^T\ \boldsymbol{a}_{k,2}^T\ \boldsymbol{a}_{k,3}^T\ \boldsymbol{a}_{k,4}^T\right]^T \text{ and } \boldsymbol{B}_{S_k} = \left[b_{k,1}\ b_{k,2}\ b_{k,3}\ b_{k,4}\right]^T, \quad (9)$$

where $\boldsymbol{a}_{k,i}^T = [I_x(x_i, y_i)\ I_y(x_i, y_i)\ \ -I(x_i, y_i)\ \ -1]$ and $b_{k,i} = I_t(x_i, y_i)$ for $i = \{1, 2, 3, 4\}$ at the kth random sampling.

Fig. 2 illustrates pictorially how a temporary solution is obtained from the pixels in a local window. This figure, a small variation on the earlier Fig. 1, uses large black dots to show the randomly selected pixels in the local window. The constraint lines corresponding to these pixels are shown thick black lines in Fig. 2b. The intersection of these thick lines corresponds to the temporary solution $\tilde{\boldsymbol{\theta}}_k$ for the example depicted in the figure. The rest of Fig. 2 is the same as Figs. 1b and 1c. The open solid circles correspond to the on-car pixels and their constraint equations are depicted as solid but thin lines in Fig. 2b. The dotted circles are pixels in the background. Finally, the double-circled pixel in the center is where we want to estimate the motion parameters.

The temporary solution $\tilde{\boldsymbol{\theta}}_k$ allows us to calculate a set of squared residuals $r^2(x, y, \tilde{\boldsymbol{\theta}}_k)$ for pixels at all the (x, y)s in the local window R. We next determine the median of squared residuals. The median is denoted by $M_k = median_{(x_i,y_i)\in R} r^2 (x_i, y_i, \tilde{\boldsymbol{\theta}}_k)$. After repeating this calculation s times for $k = \{1, 2, \ldots, s\}$, we choose a set of parameters $\tilde{\boldsymbol{\theta}}$ corresponding to the minimum median $\hat{M}$ among $\{M_1, M_2, \ldots, M_s\}$. When the total number of pixels in the local window is N, the minimization of our median would requires ${}_N C_p$ computation (i.e., the number of all possible combinations of p linear equations from all the pixels in the local window R). Obviously, the computational cost of such a naive approach would be too high to make this algorithm practical. What we need is a smaller number of samplings—as opposed to all of ${}_N C_p$ samplings—but we would want this smaller number to include at least one for which the solution to the constraint equations is correct. When N is much larger than p, the probability that s samplings would include the correct solution is given by $1 - (1 - (1 - \epsilon)^p)^s$, where ϵ is the fraction of outliers in the local window. This probability expression helps us make a more intelligent decision about giving a value to s. For example, if we wanted to include in our sampling set the correct solution with a probability of, say, 95 percent for $p = 3$ and $\epsilon = 50$, we would need to use $s = 23$.

The LMedS solution obtained from the above procedure may be inaccurate when N is small. This is a problem common to all statistical inferences that use the median. In order to improve the LMedS solution for a given N in a general statistical data analysis context, Rousseeuw and Leroy have suggested that the median-based calculation be used only to identify the data points that contribute to the outliers in the residuals and, after such outliers are removed, a LS-based approach be used to find the best solution [24]. Identifying outliers obviously requires that we first figure out what threshold to use on the residuals.

The outlier detection threshold on the residuals is expressed by the standard deviation σ of the residual error r_i. The value of σ is calculated in two steps. First, we form an initial estimate σ^0 using the minimum median $\hat{M}$ obtained from the aforementioned procedure through the following equation: $\sigma^0 = 1.4826(1 + 5/(N - p))\sqrt{\hat{M}}$, where the factor $1/\Phi^{-1}(0.75) = 1.4826$ is selected to guarantee that $median_i|z_i|/\Phi^{-1}(0.75)$ is a consistent estimator of σ when the z_is are distributed normally as $G(0, \sigma^2)$ and $\Phi(x)$ is the standard normal cdf for x. The multiplication with the correction factor $1 + 5/(N - p)$ is introduced empirically for more feasible estimations for the case of small N [24]. The initial estimate σ^0 is then used to determine an initial weight w_i for each pixel at $(x, y) = (x, y)_i$:

$$w_i = \begin{cases} 1 & \text{if } |r_i/\sigma^0| \leq 2.5 \\ 0 & \text{otherwise.} \end{cases} \qquad (10)$$

The bound is chosen to be 2.5 based on the assumption that there are few residuals larger than 2.5σ for the normally distributed data [24]. The final estimate for σ, denoted $\hat{\sigma}$, is then calculated using the data in which outliers are rejected by the initial weights as

$$\hat{\sigma} = \sqrt{\left(\sum_{i=1}^{N} w_i r_i^2\right) \Big/ \left(\sum_{i=1}^{N} w_i - p\right)}. \qquad (11)$$

The final weights for the pixels are computed by substituting σ^0 by $\hat{\sigma}$ in (10). The LS-based solution to the motion parameter vector $\hat{\boldsymbol{\theta}}$ can then be expressed as $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} w_i r_i^2$.

3.2.2 The Modified LMedS Method

The standard LMedS algorithm described in the previous section searches for a good approximate solution based on the assumption that a majority (more than 50 percent) of pixels represents a coherent motion correctly. To support this assumption, the accuracy of the brightness constraint equation is very important and it relies on the accuracy of the spatial and temporal derivatives of the image intensity. The partial image derivatives can be calculated accurately by convolving the derivative of the 3D-Gaussian function with a large number of images [9]. However, due to many practical reasons, optical flow estimation using only two consecutive image frames at a time is preferred. For an example, the most popular method for calculating the image derivatives is by using a simple discrete approximation at the center of a cube formed by eight brightness difference measurements using two consecutive image frames, as presented in [3].

When one uses the method in [3] for estimating image derivatives, greater care needs to be exercised in choosing the p pixels in each sampling from the local window for the purpose of forming a temporary solution. If possible, one should choose the pixels that are close together or have high gradient magnitudes [10]. Obviously, if the observations contain excessive noise and other artifacts, such as large illumination changes, the intersections of the constraint equations corresponding to the individual p-pixel samplings will be scattered around the correct solution and the probability of the temporary solution being correct will be low. To address this weakness of the standard LMedS method when the image derivatives are noisy, we have proposed a new approach to the construction of the temporary solution that uses more pixels than the number of constraints that need to be satisfied by the motion equations [12]. We will now briefly review this work.

The modification we proposed in [12] consists of not basing the temporary solution on p randomly selected pixels—recall that, in standard LMedS, p is the number of parameters to be estimated—but instead deriving the temporary solution from a subwindow of the local window. In our modification, the size of the subwindow is sufficiently large so that the number of pixels it contains exceeds the number of parameters to be estimated. The new temporary solution uses *all* of the pixels in the subwindow that is selected. It is now the position of the subwindow that is random inside the local window, as opposed to the choice of the individual pixels in the standard LMedS. We then solve a set of overdetermined equations using the least squares method for the temporary solution.

Our method can be better explained with the help of Fig. 3, which, except for the depiction of a subwindow inside the local window, is the same as the earlier Fig. 1 and Fig. 2. The local window is outlined by a solid square and subwindow by a dashed square. The double-circled pixel at the center of the local window is where we want to estimate the motion parameters. Instead of a random sampling of p pixels, we now work with all the pixels in the subwindow while ensuring that the subwindow size is chosen so that it contains more pixels than the number p of motion parameters. The temporary

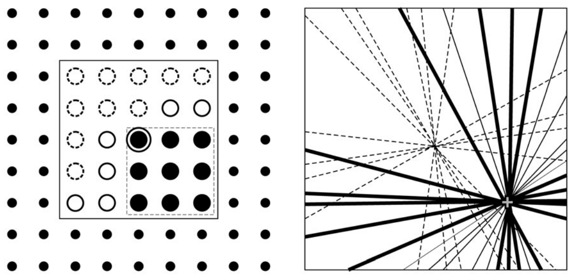


Fig. 3. (a) A local window and (b) constraint lines in the parameter space used for the modified LMedS method.

solution formed from all the pixels in a subwindow is still random, in the sense that the position of the subwindow is now random inside the local window.

In Fig. 3a, the subwindow pixels are all shown with large solid black dots. The constraint lines corresponding to these pixels are shown as thick black lines in Fig. 3b. The LS solution from the overdetermined equations corresponding to these thick lines is denoted by a cross in Fig. 3b. As before, the solid circles in the figure correspond to the on-car pixels. The constraint equations for the on-car pixels that are not in the subwindow are depicted as solid but thin lines in Fig. 3. The dotted circles are pixels in the background.

As mentioned already, the temporary solution for a given subwindow is obtained by the LS method using the overdetermined equations corresponding to the pixels in the subwindow. Since we have more equations than unknowns, we can write down the following pseudoinverse LS solution for the temporary solution for the kth subwindow: $\tilde{\boldsymbol{\theta}}_k = [\boldsymbol{A}_{S_k}^T \cdot \boldsymbol{A}_{S_k}]^{-1} \cdot \boldsymbol{A}_{S_k}^T \cdot (-\boldsymbol{B}_{S_k})$, where S_k is the set of pixels in the kth subwindow in the local window R and $\boldsymbol{A}_{S_k}$ and $\boldsymbol{B}_{S_k}$ are defined as:

$$\begin{aligned} \boldsymbol{A}_{S_k} &= \left[\boldsymbol{a}_{k,1}^T \; \boldsymbol{a}_{k,2}^T \; \dots \; \boldsymbol{a}_{k,i}^T \; \dots \; \boldsymbol{a}_{k,q}^T\right]^T \text{ and} \\ \boldsymbol{B}_{S_k} &= \left[b_{k,1} \; b_{k,2} \; \dots \; b_{k,i} \; \dots \; b_{k,q}\right]^T, \end{aligned} \tag{12}$$

where $\boldsymbol{a}_{k,i}^T = [I_x(x_i, y_i) \; I_y(x_i, y_i) \; -I(x_i, y_i) \; -1]$ and $b_{k,i} = I_t(x_i, y_i)$ for $i = \{1, 2, \dots, q\}$ in the kth subwindow.

For each temporary solution, the residual error and the median of the squared residual errors are calculated for all the pixels in the local window R. The final solution $\hat{\boldsymbol{\theta}}$ is then obtained using the weighted LS method as described in the previous section.

4 A Baseline Implementation of the LMedS Method that Uses All Pixels

The main focus of this paper is the comparative error analysis of the two LMedS methods presented in Sections 3.2.1 and 3.2.2. We also want the accuracy of the two methods to be compared to some sort of a "gold standard." Since both methods use random samplings in a local window to form motion estimates in the center of the window, our error analysis must focus especially on any deleterious effects of random sampling. As the reader will recall, the random sampling in both methods is based on the assumption that the probability of one of the samplings incorporating the correct solution is close to 1. While, as we showed in our discussion in

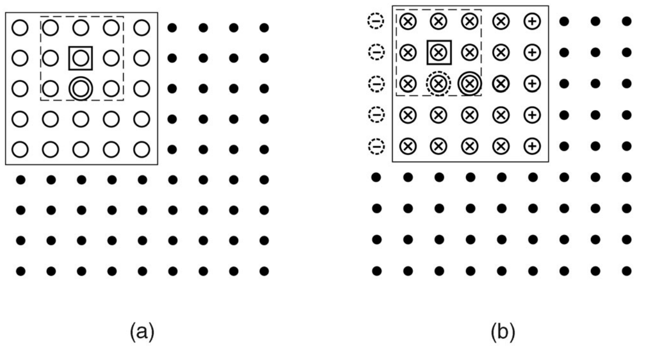


Fig. 4. Local windows used for a baseline implementation of the LMedS method.

Section 3.2.1, it is possible to theorize about this probability, in practice, it is difficult to ensure that a chosen number of samplings in a local window would satisfy the constraint on the probability.

We therefore believe that a "gold standard" for studying the error properties of LMedS-based approaches must include all the pixels in a local window. This is exactly what we have done with our baseline implementation of the LMedS approach—it uses all the pixels in a local window to form the motion estimates at the center of the window. As one would expect, a naive LMedS implementation that uses all pixels would be much too slow computationally to be of much use. To speed things along in our baseline implementation of LMedS, we decided to use the "running estimation" idea of Huang et al. in [17]. We will next describe this implementation.

Fig. 4 illustrates pictorially how the running technique is employed in our baseline implementation of the LMedS method. Fig. 4a shows a 9×9 segment of an image frame and a 5×5 local window in it that is outlined with a black square. As before, for each such window we estimate the motion at the central double-circled pixel. Note, in this baseline implementation, the centers of the subwindows are positioned at *every* pixel in the local window rather than at a small number of randomly selected pixels in the previous two LMedS methods. For each subwindow, a temporary solution $\tilde{\boldsymbol{\theta}}$ is calculated using the LS method with the overdetermined constraint equations employing all the pixels in the subwindow. With this temporary solution, the squared residual error $r^2(x, y, \tilde{\boldsymbol{\theta}})$ is calculated for every pixel (x, y) in the local window. The values of the temporary solution and residual errors from each subwindow are stored in the memory for later use. Fig. 4b shows the local window moved to the next position with a rightward shift of one pixel. The pixels denoted by the dotted circles with a "$-$" sign represent those that belong to the old local window but not to the new shifted one, the pixels denoted by the solid circles with a "$+$" sign represent those that do not belong to the old window but belong to the new local window, and the pixels denoted by the sold circles with a "$\times$" sign represent those that are common to both local windows. An important observation to be made here is that the old-window temporary solutions and residual errors at the common pixels can be reused in the new window. For example, the temporary solution for the pixel enclosed by a small square in Fig. 4b is the same as that for the pixel enclosed by a small square in Fig. 4a and the residual errors with this temporary solution for the common pixels in the new window are the same as the values that are calculated in the old window.

In order to present the running median algorithm, we must now use a two-dimensional index for the temporary solutions derived from the subwindows within a local window. But, that entails that we must also use double indices for designating the coefficient matrices A and B that are derived from the constraint equations in each subwindow.

Consider a subwindow that is centered at the offset (m, n) with respect to the local-window whose central pixel is at (x, y). We will denote the temporary solution for this subwindow by $\tilde{\boldsymbol{\theta}}_{(m,n)}(x, y)$. For example, for the pixel enclosed by a small square that is at the center of the subwindow enclosed by a large dashed square in Fig. 4a, $m = 0$ and $n = -1$ and the temporary solution for this pixel can be represented by $\tilde{\boldsymbol{\theta}}_{(0,-1)}(x, y)$ for the local window centered at (x, y). The temporary solution is obtained by the LS method with the overdetermined constraint equations using the pixels in the subwindow centered at $(x + m, y + n)$ as follows:

$$\tilde{\boldsymbol{\theta}}_{(m,n)}(x, y) = \left(\boldsymbol{A}^T_{S_{(m,n)}} \cdot \boldsymbol{A}_{S_{(m,n)}}\right)^{-1} \cdot \boldsymbol{A}^T_{S_{(m,n)}} \cdot \left(-\boldsymbol{B}_{S_{(m,n)}}\right), \quad (13)$$

where $S_{(m,n)}$ is the set of pixels in the subwindow centered at $(x + m, y + n)$ in the local window R centered at (x, y).

In (13),

$$\boldsymbol{A}_{S_{(m,n)}} = \left[\boldsymbol{a}^T_{(m,n),1} \ \cdots \ \boldsymbol{a}^T_{(m,n),i} \ \cdots \ \boldsymbol{a}^T_{(m,n),q}\right]^T$$

and

$$\boldsymbol{B}_{S(m,n)} = \left[b_{(m,n),1} \ \cdots \ b_{(m,n),i} \ \cdots \ b_{(m,n),q}\right]^T,$$

where

$$\boldsymbol{a}^T_{(m,n),i} = \boldsymbol{a}^T(x_i, y_i) = [I_x(x_i, y_i) \ I_y(x_i, y_i) - I_t(x_i, y_i) - 1],$$

and $b_{(m,n),i} = b(x_i, y_i) = I_t(x_i, y_i)$ for $i = \{1, 2, \ldots, q\}$ in the subwindow centered at $(x + m, y + n)$ in the local window centered at (x, y). Note that q is the total number of pixels in a subwindow. The residual error with the temporary solution $\tilde{\boldsymbol{\theta}}_{(m,n)}$ for the pixel at (x_i, y_i) in the local window R centered at (x, y) may now be expressed by the form

$$r(x_i, y_i, \tilde{\boldsymbol{\theta}}_{(m,n)}(x, y)) = \boldsymbol{a}^T(x_i, y_i) \cdot \tilde{\boldsymbol{\theta}}_{(m,n)}(x, y) + b(x_i, y_i). \quad (14)$$

If the old local window is centered at (x, y) and the new local window is centered at $(x + k, y + l)$, the temporary solution and the residual errors for the common pixels in the new local window can be obtained from the pre-calculated values in the old local window as follows:

$$\tilde{\boldsymbol{\theta}}_{(m,n)}(x + k, y + l) = \tilde{\boldsymbol{\theta}}_{(m+k,n+l)}(x, y), \quad (15)$$

$$r^2(x_i, y_i, \tilde{\boldsymbol{\theta}}_{(m,n)}(x + k, y + l)) = r^2(x_i, y_i, \tilde{\boldsymbol{\theta}}_{(m+k,n+l)}(x, y)), \quad (16)$$

where (x_i, y_i) are the pixel coordinate for a common pixel.

For the implementation of the running-median-based algorithm, an image frame is covered by scanning it with a local window in the following manner. A local window first moves from left to right. When it hits the right edge of the image frame, it moves down. Subsequently, it moves from right to left, etc. This implies three different types of shifts for the local window—shift to the right, shift downward, and shift to the left. For these three types of shifts, the values of (k, l) in (15) and (16) are $(1, 0)$, $(0, 1)$, and $(-1, 0)$, respectively.

At every shift of the local windows as described above, a median of the squared residual errors is calculated for each subwindow in the local window. Since the residual errors for the common pixels in the new local window are the same as the precalculated values in the old local window, for the median calculation in the new local window, we just need to update the set of squared residual errors that was used in the old local window. This updating is done by adding the newly calculated squared residual errors for the newly added pixels to, and removing the values for the expired pixels from, the set of squared residual errors that was used in the old local window. When the old local window is centered at (x, y) and the new local window is centered at $(x + k, y + l)$, the median of the squared residual errors for the subwindow centered at the offset (m, n) with respect to the center of the new local window can be formally stated as follows:

$$\begin{aligned} &M_{(m,n)}(x+k, y+l) \\ &= \underset{(x_i,y_i)\in R(x+k,y+l)}{median} r^2(x_i, y_i, \tilde{\boldsymbol{\theta}}_{(m,n)}(x+k, y+l)) \qquad (17)\\ &= median\big\{\{r^2(x_i, y_i, \tilde{\boldsymbol{\theta}}_{(m+k,n+l)}(x,y))|(x_i,y_i) \in R(x,y)\} \\ &\quad - \{r^2(x_i, y_i, \tilde{\boldsymbol{\theta}}_{(m+k,n+l)}(x,y))|(x_i,y_i) \in R^-(x,y)\} \\ &\quad + \{r^2(x_i, y_i, \tilde{\boldsymbol{\theta}}_{(m,n)}(x+k,y+l))|(x_i,y_i) \\ &\quad \in R^+(x+k, y+l)\}\big\}, \qquad (18) \end{aligned}$$

where $R(x, y)$ is a set of pixels in the old local window centered at (x, y), $R^-(x, y)$ is a set of expired pixels in the old local window, $R^+(x + k, y + l)$ is a set of newly added pixels in the new local window.

The medians of the residual errors needed for each of the local windows are calculated most efficiently with the histogram-based method proposed by Huang et al. in [17]. Since this method uses integer data, we must first quantize the residual errors in each local window. We will now present a residual error normalization procedure that results in good normalization of the range of the residual errors. Ordinarily, as our following discussion illustrates, the residual error at a pixel is directly proportional to the magnitude of the image gradients at that pixel. So, a convenient normalization consists of dividing out the residuals by the size of the gradients.

As we described in Section 3, the optical flow constraint equation can be represented by a line in the (u, v) parameter space. It is interesting to geometrically examine a temporary solution obtained from one subwindow vis-a-vis the residual errors corresponding to this solution at all the pixels in the local window. Consider the case presented in Fig. 5 where the line depicts the constraint equation for some pixel P at (x, y) for the temporary solution $(\tilde{u}, \tilde{v})$ that was obtained for the first subwindow at the upper left corner of a local window. We now investigate the relationship between the residual error given by the value of $I_x(x, y)\tilde{u} + I_y(x, y)\tilde{v} + I_t(x, y)$ at P and the distance d from $(\tilde{u}, \tilde{v})$ to the line as shown in Fig. 5. The distance d in Fig. 5 can be easily calculated as follows:

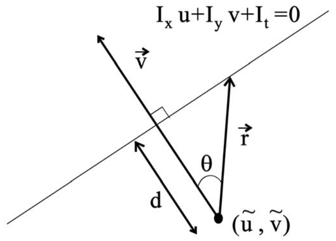


Fig. 5. Distance from a temporary solution to a constraint line in the two-dimensional parameter space.

$$\begin{aligned} d = |\vec{r}|\cos\theta \; = \; |\hat{n}\cdot\vec{r}| \; &= \left| \frac{(I_x, I_y)}{\sqrt{I_x^2 + I_y^2}} \cdot (u - \tilde{u}, v - \tilde{v}) \right| \\ &= \frac{|I_x\tilde{u} + I_y\tilde{v} + I_t|}{\sqrt{I_x^2 + I_y^2}}, \end{aligned} \qquad (19)$$

where $\vec{r}$ is a vector from $(\tilde{u}, \tilde{v})$ to any arbitrary point on the line, $\vec{n} = (I_x, I_y)$ is a vector that is perpendicular to the line, $\hat{n}$ is a unit vector having the same direction of $\vec{n}$, and θ is the angle between the vector $\vec{n}$ and $\vec{r}$. Now, the squared residual error E can be represented as follows:

$$E^2 = (I_x\tilde{u} + I_y\tilde{v} + I_t)^2 = (I_x^2 + I_y^2)\cdot d^2. \qquad (20)$$

As we can see in (20), the magnitude of the squared residual errors is proportional to the magnitude of the sum of the squared image gradients, $I_x^2 + I_y^2$, for a given value of the distance d. This dependency of the residual error on the image gradient is not desirable for a histogram-based method for median calculation because the data range will vary as the local window shifts in an image frame. Therefore, we employ the distance d that does not depend on the image gradient as the other type of the residual error and call it as the normalized residual error.

To show the advantages of normalizing the residual errors in the manner indicated, we show in Figs. 6c and 6d the histograms of the residual errors, the former unnormalized and the latter normalized, for the case of the local window shown with a white box in the upper left quadrant of the image of Fig. 6a. This local window corresponds to the case of low image intensity gradients. As the reader can see, the histogram is highly lop-sided for the nonnormalized residual errors, but reasonably uniform for the normalized residual errors. Fig. 6b shows the constraint equations for all of the pixels in the local window, the ground-truth values for the motion parameters are shown with a white cross in the figure. Figs. 7a, 7b, 7c, and 7d show the same for the case of local window at the lower right corner of the image frame. This example corresponds to the case of high local image gradients. We show this second example to illustrate that the histograms of the nonnormalized residual errors can vary considerably from local window to local window and their wide variation precludes any simple approach to quantization. Hence, we must resort to error normalization.

5 Error Analysis

As mentioned at the outset, the main goal of this paper is to carry out a comparative error analysis of the following two LMedS optical-flow estimation methods that incorporate a

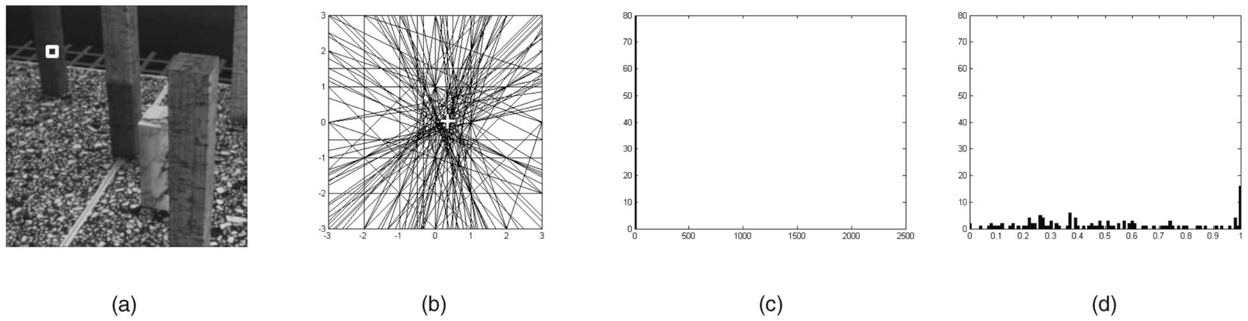


Fig. 6. (a) An image frame and a local window, (b) constraint lines in a parameter space, (c) histogram of unnormalized residual errors, and (d) histogram of normalized residual errors.

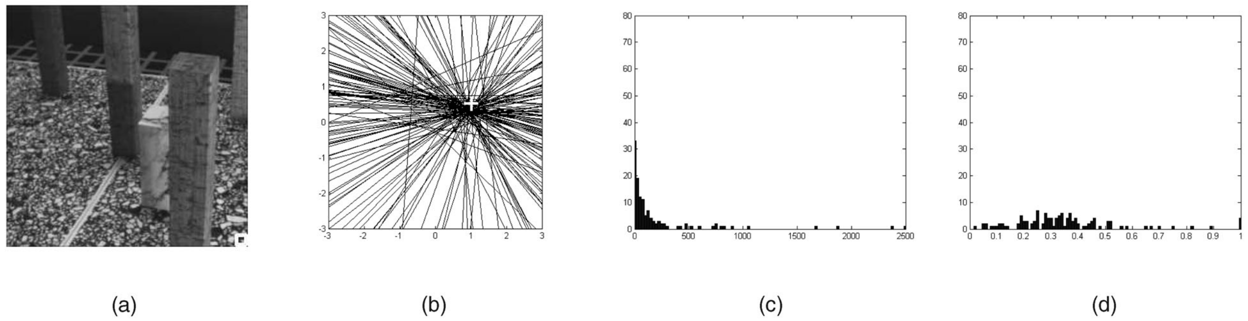


Fig. 7. (a) An image frame and a local window, (b) constraint lines in a parameter space, (c) histogram of unnormalized residual errors, and (d) histogram of normalized residual errors.

model for varying illumination: 1) a method that uses the standard LMedS regression as presented by Rousseeuw and Leroy in [24] and 2) its modified version as presented by us in [12]. The goal of this paper is to also compare these two methods with the gold-standard implementation described in Section 4.

Generally, performance characterization of a computer vision algorithm has to do with the accuracy of the output vis-a-vis random variations and imperfections in the input [29]. In most performance characterization studies, two types of random perturbations are introduced into the input: small perturbations that affect all data units and large perturbations that affect only a small fraction of the data units [29].

Following this established practice, the performance characterization study in this paper also injects two forms of noise into the video sequences used for motion estimation: We add small random fluctuations to all of the recorded images in a video sequence in order to simulate the discrete nature of the noise in the photocell detectors of a camera. This type of noise is generally modeled by a zero mean Gaussian process and quantified by different variances. The other type of noise we inject into input data consists of drop-out noise, also commonly referred to as the salt-and-pepper noise. This type of noise, caused by errors in data transmission, corrupts the pixels by changing their values to either the maximum possible value or to the zero value, thus giving the image a "salt and pepper" appearance. The rest of the pixels remain unaffected by this type of noise. This type of noise is usually quantified by the density of the pixels which are corrupted.

A performance characterization of the sort we present here must of necessity calculate errors between the ground-truth values, on the one hand, and the estimated values, on the other, for the motion vectors. What makes our task particularly difficult is that our ground-truth data must be for the case when motion is recorded under varying illumination. There do exist publically available ground-truthed image sequences for motion-estimation experiments, but unfortunately they do not come with illumination variations. For example, there is the constant-illumination ground-truthed image sequence made available by Otte and Nagel that was recorded by a camera mounted on a robot arm moving with a precalculated trajectory [30]. The scene for this data consisted of polyhedral objects. The ground-truth value for the motion vectors is calculated from the known trajectory of the camera. Another ground-truthed image sequence, also for polyhedral objects and also for the case of constant illumination, has been made available by McCanne et al. [31]. The ground truth information, in this case, is obtained by first manually assigning motion vectors to the corners of polyhedral objects and then estimating the rest of the motion vectors assuming a projective linear motion model for the pixels inside each polyhedral face.

We believe we have four options for generating the sort of ground-truthed data we need for our error analysis:

1. Use a synthetic image sequence with artificial illumination variation.
2. Use an existing publically available image sequence taken under constant illumination and then inject artificial illumination variation into it.
3. Use a graphics package such as OpenGL to create a synthetic video clip with multiple motions and frame-to-frame illumination variations.
4. Record a new image sequence with real illumination variations and make that publically available.

The experimental results we report in this section use all four of these approaches for error evaluation, giving us a

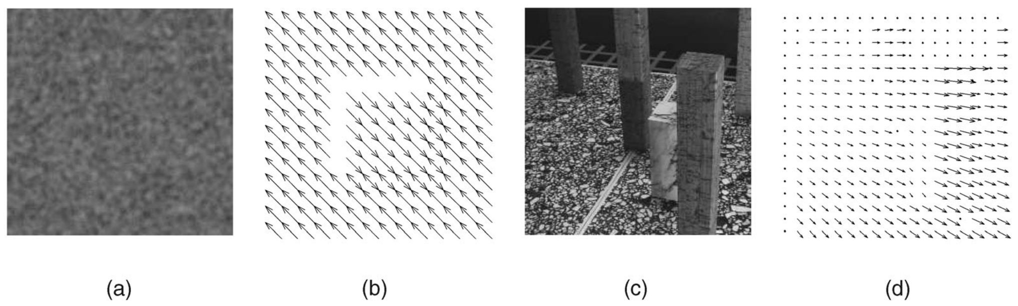


Fig. 8. (a) The first image frame of the random-dot sequence (b) its ground-truth motion, (c) the first image frame of the modified Otte sequence, and (d) its ground-truth motion.

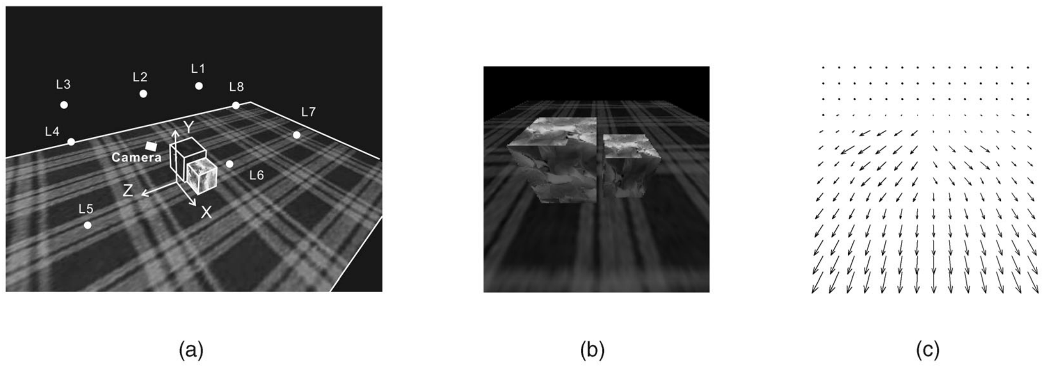


Fig. 9. (a) Camera and Lighting setup for the nine-frame OpenGL-generated sequence, (b) the first frame of this sequence, and (c) its ground-truth motion.

good mix between synthetic data and real data. Here, is a summary of the data sets used:

1. A synthetic two-frame random-dot image sequence including illumination variation.
2. A modified version of the real "Otte" image sequence [30]. The modification consists of injecting an artificial illumination variation into the data. This sequence also consists of two frames.
3. A **nine-frame** video sequence generated using OpenGL. The use of OpenGL allows us into incorporate arbitrary object motions and illumination variations into the sequence.
4. A two-frame **real** image sequence with real illumination variations that was ground-truthed for the results shown here.

With regard to the synthetic random-dot images, the first image is shown in Fig. 8a. The second image frame for this image sequence is created by translating the pixels on a centered square of the first image frame by one pixel to the right and by one pixel down and translating the background pixels by one pixel to the left and by one pixel up. For the artificial illumination variation, the intensity of pixels in the second image frame is multiplied by a factor that varies linearly from the center of the image to the corner of the image radially (1.25 at the center and 0.75 at the corner), an offset (of 10) is then added to the result. Fig. 8b shows the ground-truth optical flow at the sampled pixels of the image sequence.

For the second data set for our error analysis, the first frame of the modified "Otte" sequence is shown in Fig. 8c. Instead of using the original "Otte" sequence used in [30], we reduced the size of the original images to half and then we also injected the artificial illumination variation that is the same as used in the random-dot image sequence. The ground-truth optical flow for this sequence is shown in Fig. 8d.

As mentioned already, our third data set for error analysis uses nine ground-truthed frames generated by the OpenGL-based modeling software. This data includes a much more complicated illumination variation than is the case with the first two data sets. Fig. 9 a shows a 3D view of the synthetic scene used for OpenGL. It consists of two boxes sitting on top of a textured surface. Overlaid on the scene are the camera position (but note that the camera is allowed to move frame-to-frame) and the positions of the eight light sources used for creating illumination variations. The light sources can be turned on or off individually on a frame-to-frame basis and, when a light source is on, its intensity can be varied continuously. Fig. 9b is the first image frame of the synthetic video sequence. The ground-truth optical flow for the synthetic image sequence can be calculated from the known 3D points corresponding to all the image pixels and projecting the 3D motion of the points on to the 2D image plane. Fig. 9c is the ground-truth optical flow for the first image frame of the sequence. Table 1 shows the 3D coordinates of the centers of the two boxes and the optical center of the camera with respect to the base coordinate frame shown in Fig. 9a and the on-off sequence for each of the eight light sources. The two boxes move laterally along the "X" axis and the camera moves up and down along the "Y" axis or back and forth along the "Z" axis. Until the sixth image frame (frame number 5), all the lights are turned on, but their intensity varies frame to frame, as shown in the table. After the sixth frame, one of the lights sources is turned off for each of the remaining frames.

TABLE 1
Position of the Boxes and the Camera for the Nine-Frame OpenGL-Generated Sequence, the On-Off Pattern of the Illumination Sources, and the Intensity of the On Sources

frame	left box			right box			camera											Intensity
number	x	y	z	x	y	z	x	y	z	L1	L2	L3	L4	L5	L6	L7	L8	of lights
0	-20.0	0.0	-30.0	20.0	0.0	-30.0	0.0	65.0	36.0	ON	ON	ON	ON	ON	ON	ON	ON	0.015
1	-20.2	0.0	-30.0	20.2	0.0	-30.0	0.0	65.0	35.5	ON	ON	ON	ON	ON	ON	ON	ON	0.020
2	-20.4	0.0	-30.0	20.4	0.0	-30.0	0.0	65.0	35.0	ON	ON	ON	ON	ON	ON	ON	ON	0.025
3	-20.6	0.0	-30.0	20.6	0.0	-30.0	0.0	65.0	35.5	ON	ON	ON	ON	ON	ON	ON	ON	0.030
4	-20.8	0.0	-30.0	20.8	0.0	-30.0	0.0	65.5	36.0	ON	ON	ON	ON	ON	ON	ON	ON	0.035
5	-21.0	0.0	-30.0	21.0	0.0	-30.0	0.0	66.0	36.0	ON	ON	ON	ON	ON	ON	ON	ON	0.040
6	-21.2	0.0	-30.0	21.2	0.0	-30.0	0.0	65.5	36.0	ON	ON	OFF	ON	ON	ON	ON	ON	0.040
7	-21.4	0.0	-30.0	21.4	0.0	-30.0	0.5	65.0	36.0	ON	ON	ON	OFF	ON	ON	ON	ON	0.040
8	-21.6	0.0	-30.0	21.6	0.0	-30.0	1.0	65.0	36.0	ON	ON	ON	ON	OFF	ON	ON	ON	0.040

For the final data set for our error analysis, a real image sequence was recorded using a setup shown schematically in Fig. 10a. There are two light sources A and B of different output spectra in the setup. Light source A, consisting of multiple fluorescent lamps mounted in the ceiling, was used for recording the first image of an image pair, and light source B, an incandescent lamp, was added to the light source A for the second image. Since the two light sources were not physically colocated they obviously produced different shadow patterns in the scene. With regard to the motions in the scene, the objects P and Q were shifted laterally to the left and to the right, respectively. Using our software tool based on the same rationale that was presented in [31], we calculated the ground truth motion data from the two image frames. The first frame of the two images thus recorded is shown in Fig. 10b. Shown in Fig. 10c is a pictorial depiction of the ground-truth velocities. The horizontal components of these velocities at each of the pixels are shown in Fig. 10d.

Optical flow was estimated from all four data sets using the algorithms listed in Table 2. For methods that use regularization parameters, we used two versions, with the parameters all set to 1 and with the parameters estimated optimally in the manner indicated earlier. The method names are starred for the versions when the regularization parameters were estimated optimally. The table also shows where a method is based on obtaining a global solution or local solution, and whether or not a method is designed to accommodate varying illumination.

This is how we have organized the presentation of the error evaluation results:[2] First, we will show optical flow estimation errors qualitatively, but only for the random-dot synthetic space. Next, we will describe how we measure the optical-flow estimation error quantitatively. Subsequently, we will present comparative results for each of the four data sets.

Fig. 11 shows the qualitative optical flow results for the random-dot image sequence. From this figure, we can obviously see that the methods that are designed to work with only constant illumination produce poor estimates of optical flow. We can also see that our global (RVL-G, RVL-G*) and local integrated methods (RVL-L1, RVL-L2) that solve both the varying illumination and the motion discontinuity problems simultaneously provide accurate optical flow results especially at the motion boundaries. Additionally, it can also be seen that the global methods perform poorly when the regularization parameters are not optimized (see Fig. 11c and Fig. 11e).

For quantitative error analysis, following [32] we calculated the angular error at each pixel by measuring the angle between the 3D normalized versions of the correct optical flow vector and the estimated optical flow vector. For this, the normalized 3D vector is defined as $\vec{v} = \frac{1}{\sqrt{u^2+v^2+1}}(u, v, 1)^T$ and the angular error as $E = \arccos(\vec{v_c}, \vec{v_e})$, where the normalized 3D vector $\vec{v_c}$ corresponds to the correct velocity and $\vec{v_e}$ to the estimated velocity.

Table 3 shows the average angular error and its variance obtained with the different methods when applied to the random-dot sequence for three different cases: no noise, Gaussian noise at three different levels, and salt-and-pepper noise, again at three different levels. The dependence of the

2. Space considerations and desire to impart maximum possible information have dictated this organization of the experimental results.

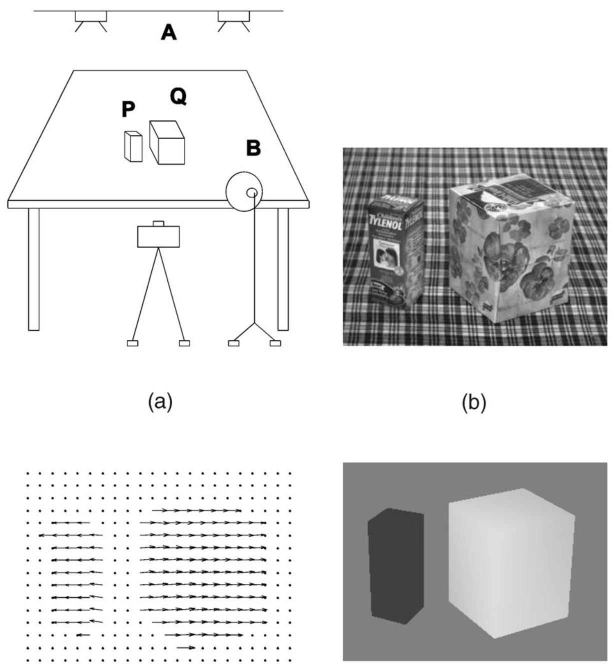


Fig. 10. (a) Camera and lighting setup for the real image sequence, (b) the first frame and (c), (d) the ground-truth motion.

TABLE 2
Methods Used for Error Analysis

Methods	Abbrev.	Global vs. Local Method	Constant vs. Varying Ill. model
Horn & Schunck [3] with untuned parameter (set to 1)	HS	Global	Constant
Horn & Schunck [3] with optimal parameter	HS*	Global	Constant
Black & Anandan [2] with untuned parameter (set to 1)	BA	Global	Constant
Black & Anandan [2] with optimal parameter	BA*	Global	Constant
Gennert & Negahdaripour [6] with untuned parameters (set to 1's)	GN	Global	Varying
Gennert & Negahdaripour [6] with optimal parameters	GN*	Global	Varying
Our global integrated method [13] with untuned parameters (set to 1's)	RVL-G	Global	Varying
Our global integrated method [13] with optimal parameters	RVL-G*	Global	Varying
Lucas & Kanade [5]	LK	Local	Constant
The standard LMedS [24], [10], [9]	SLMS	Local	Constant
The modified LMedS [12]	MLMS	Local	Constant
Negahdaripour & Yu [23]	NY	Local	Varying
Our integrated method based on the standard LMedS [12]	RVL-L1	Local	Varying
Our integrated method based on the modified LMedS [12]	RVL-L2	Local	Varying

average angular error on the noise level is also shown graphically in Fig. 12 for each of the different methods. From the results shown, it is obvious that the methods that use the constant illumination model produce large errors. On the other hand, our integrated global method (RVL-G*) shows superior results overall. However, note at the same time the poor results produced by our method RVL-G that uses untuned values for the regularization parameters. When noise levels are low, our integrated local method based on the modified LMedS technique (RVL-L2) produces accuracies comparable to those obtained with the global integrated method with optimal parameters (RVL-G*). It is also interesting to note that, in the presence of noise, our global integrated method outperforms the other methods for the case of Gaussian noise and our local integrated method does the same for the case of salt-and-pepper noise.

Table 4 and Fig. 13 show the results obtained for the modified Otte sequence that, as mentioned previously, consists of more realistic images. The overall conclusions to be drawn from this table and the figure are essentially the same as for Table 3 and Fig. 12.

We will now demonstrate results, very similar to those already shown, with the help of the nine-frame OpenGL generated (and ground-truthed) sequence of images taken under varying illumination. Figs. 14a, 14b, and 14c show the average angular error for each image frame of this sequence with no noise, with Gaussian noise, and with salt-and-pepper noise, respectively. For the Gaussian noise, we used 4 for the variance, and for the salt-and-pepper noise, a density of 0.01. For the results shown, the regularization parameters for the global methods are tuned only for the first image frame and then kept the same for the remaining frames. As can be seen in Fig. 14a, the errors produced by the methods that use the constant illumination model are much larger than those of the methods based on the varying illumination model for all the image frames. The global and local integrated methods, RVL-G, RVL-L1, and RVL-L2, again outperform the other methods. However, the global method RVL-G performs worse than the local methods RVL-L1 and RVL-L2 after the first frame because of the sensitivity of the regularization parameters. Fig. 14b shows that the performance of all the integrated methods deteriorated at almost the same level of noise for the case of Gaussian noise. Also, Fig. 14c demonstrates the superior robustness of RVL-L1 and RVL-L2 to the salt-and-pepper noise for all the image frames. It is interesting to note that RVL-L1 is not as robust as RVL-L2.

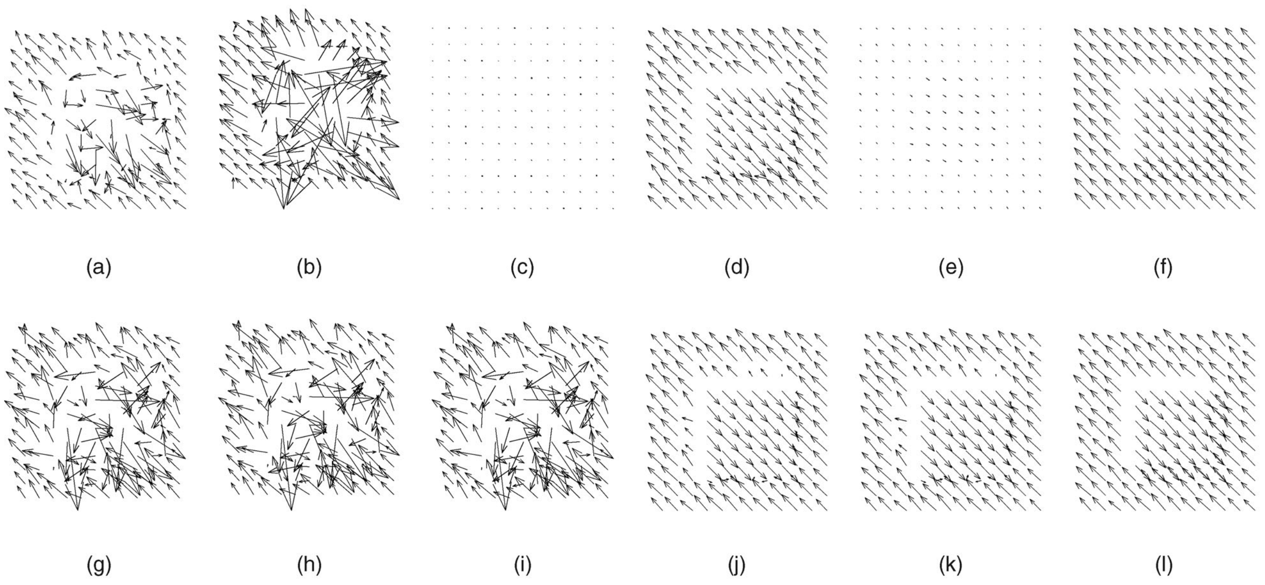


Fig. 11. Qualitative results obtained by different optical flow methods on the random-dot image sequence. (a) HS. (b) BA. (c) GN. (d) GN^*. (e) RVL-G. (f) $\mathrm{RVL\text{-}G}^*$. (g) LK. (h) SLMS. (i) MLMS. (j) NY. (k) RVL-L1. (l) RVL-L2.

TABLE 3
Average Angular Errors and Variances for the Random-Dot Sequence

Method	Ave. angular error / variance without noise	Ave. angular error / variance with different variances of Gaussian noise			Ave. angular error / variance with different densities of salt-and-pepper noise		
		2	4	8	0.005	0.01	0.02
HS*	21.26/18.19	23.07/17.86	24.82/17.75	28.04/17.20	27.17/18.62	31.98/19.17	38.89/19.01
BA*	31.37/33.95	33.24/32.57	35.02/34.15	36.08/34.60	34.60/30.54	34.31/30.25	34.52/26.73
GN	52.07/ 0.93	52.09/ 0.93	52.13/ 0.95	52.20/ 1.05	52.06/ 8.56	51.92/11.33	52.28/15.38
GN*	5.43/ 7.62	16.90/ 8.96	24.02/10.03	32.11/10.92	10.96/13.08	16.08/16.03	24.94/19.33
RVL-G	48.43/ 3.78	50.67/ 2.62	52.25/ 1.57	54.03/ 1.62	52.19/ 2.56	53.97/ 1.86	54.71/ 1.38
RVL-G*	2.27/ 8.09	2.79/ 8.38	3.36/ 8.07	4.47/ 9.57	4.64/13.65	7.45/17.55	12.83/23.07
LK	26.69/25.91	26.89/24.78	27.77/23.88	29.93/21.81	37.67/20.14	42.63/16.69	48.13/10.88
SLMS	20.87/22.46	21.23/21.31	22.52/20.60	25.24/18.81	36.52/21.91	41.01/19.86	47.16/16.51
MLMS	27.41/26.15	27.53/25.02	28.34/24.18	30.37/22.10	34.53/21.60	38.84/19.18	44.04/14.97
NY	6.64/10.95	10.85/10.94	15.13/10.58	22.01/10.26	40.85/22.57	49.15/17.35	56.44/13.19
RVL-L1	6.64/10.99	10.81/10.97	15.04/10.62	21.90/10.29	25.44/23.82	32.01/24.79	39.30/25.46
RVL-L2	4.18/ 8.73	9.18/ 9.18	13.93/ 8.73	21.61/ 9.64	4.39/ 9.34	4.61/ 9.47	6.64/12.33

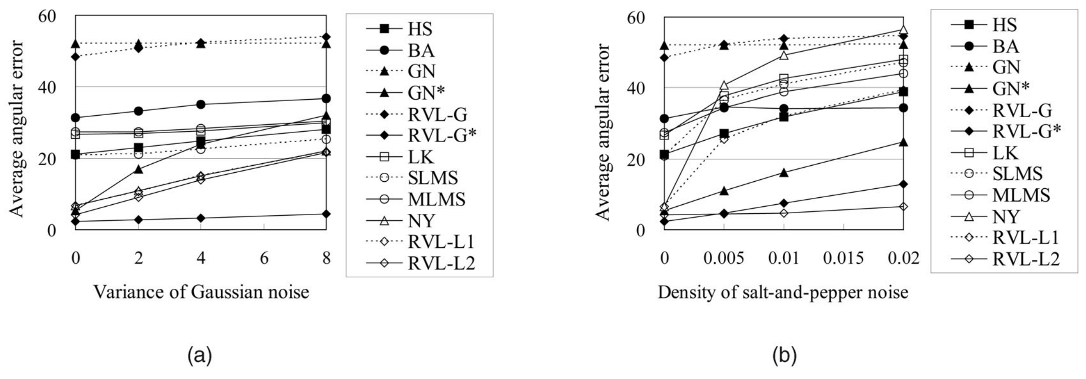


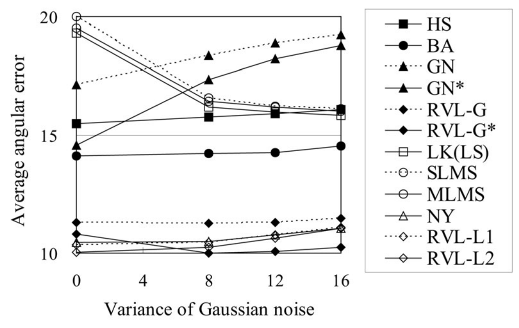
Fig. 12. Average angular errors for the random-dot sequence with (a) Gaussian noise and (b) salt-and-pepper noise.

TABLE 4
Average Angular Errors and Variances for the Modified Otte Sequence

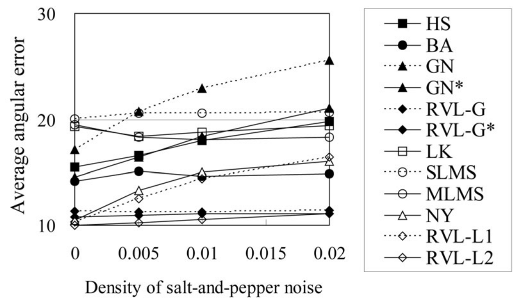
Method	Ave. angular error / variance without noise	Ave. angular error / variance with different variances of Gaussian noise			Ave. angular error / variance with different densities of salt-and-pepper noise		
		8	12	16	0.005	0.01	0.02
HS*	15.46/12.48	15.75/12.20	15.90/12.11	16.09/12.01	16.57/12.21	17.99/12.55	19.82/12.98
BA*	14.12/ 7.19	14.21/ 7.05	14.23/ 6.91	14.52/ 6.75	15.13/ 6.78	14.64/ 6.90	14.84/ 6.65
GN	17.13/ 9.76	18.36/ 9.97	18.87/10.11	19.24/10.27	20.74/12.38	22.97/13.61	25.62/15.12
GN*	14.56/ 7.63	17.32/ 9.30	18.22/ 9.89	18.78/10.36	16.47/ 9.72	18.39/11.32	21.08/13.40
RVL-G	11.30/ 6.05	11.26/ 5.28	11.30/ 5.36	11.48/ 5.47	11.24/ 6.25	11.29/ 6.57	11.39/ 6.57
RVL-G*	10.80/ 6.94	10.00/ 6.05	10.09/ 6.13	10.26/ 6.23	10.92/ 7.19	11.07/ 7.94	11.08/ 8.27
LK	19.30/17.51	16.17/14.07	15.96/13.59	15.84/13.20	18.42/14.31	18.78/13.17	19.39/12.77
SLMS	20.01/18.41	16.58/14.46	16.23/13.98	16.11/13.74	20.61/16.96	20.62/16.20	20.70/14.64
MLMS	19.52/17.64	16.41/14.24	16.16/13.77	16.01/13.34	18.29/14.50	18.07/13.39	18.34/13.00
NY	10.45/ 7.02	10.48/ 6.03	10.76/ 6.13	11.04/ 6.23	13.32/ 9.11	15.04/ 9.60	16.06/ 9.96
RVL-L1	10.34/ 6.66	10.44/ 5.83	10.76/ 6.01	11.07/ 6.17	12.49/ 8.56	14.41/ 9.83	16.41/11.13
RVL-L2	10.03/ 6.51	10.23/ 5.61	10.62/ 5.95	11.04/ 6.24	10.23/ 6.91	10.54/ 7.45	11.09/ 8.27

Since it is obvious from the results already shown that the local integrated methods are substantially superior to the other methods, we will focus exclusively on such methods for the next data set, which consists of the *real* ground-truthed image sequence with *real* illumination variations. Fig. 10, shown previously, illustrated the experimental setup used for recording the data for this image sequence. That also brings us to the main goal of this paper as stated in the opening paragraph of this section: It is to carry out a comparative error analysis of the two LMedS-based optical-flow estimation methods RVL-L1 and RVL-L2 and to further compare these two methods with the gold standard we described in Section 4. We will refer to the gold standard method by the abbreviation "Base" (for "baseline").

Table 5 shows the average angular error for the real data with the RVL-L1, RVL-L2, and the gold-standard Base methods for three cases of Gaussian noise and for three cases of salt-and-pepper noise. We show the results

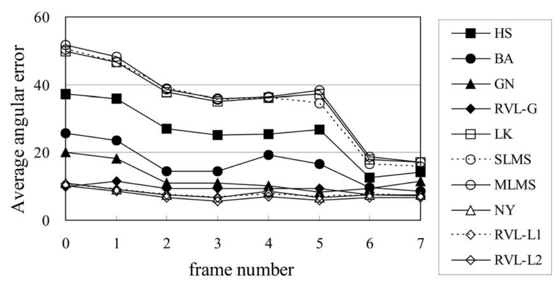


(a)

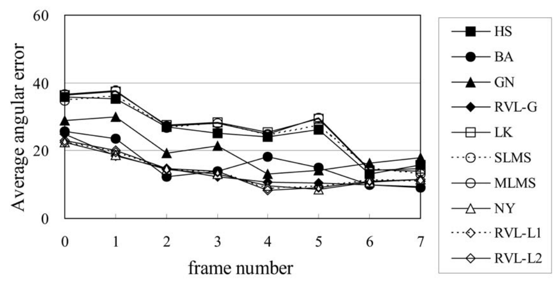


(b)

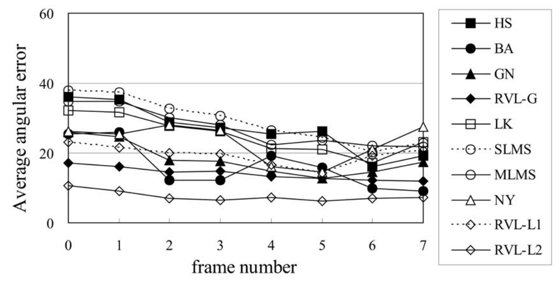
Fig. 13. Average angular errors for the modified Otte sequence with (a) Gaussian noise and (b) salt-and-pepper noise.



(a)



(b)



(c)

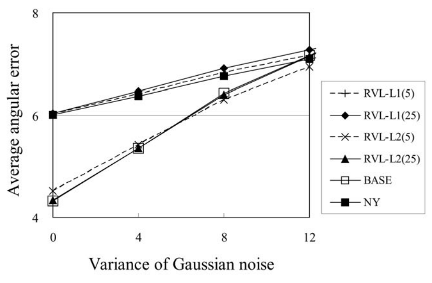
Fig. 14. Average angular error versus image frames for Synthetic-boxes sequence (a) without noise (b) with Gaussian noise, and (c) with salt-and-pepper noise.

separately for two different versions of the RVL-L1 and RVL-L2 algorithms: When only five random samplings are used for forming temporary solutions and when 25 random samplings are used. For better visualization, the same results are shown graphically in Figs. 15a and 15b. The number of random samplings used for RVL-L1 and RVL-L2 are shown parenthesized after the labels. Just to point out how much better the median-based approaches are compared to the least-squares-based approaches, the last row of Table 5 and one of the curves in Fig. 15a corresponds to the case when the motion vectors were estimated using the approach of Negahdaripour and Yu in [23]. These results are labeled "NY" in both the table and the figure.

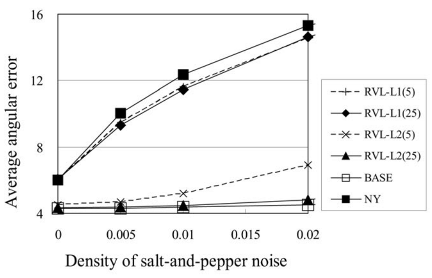
Table 5 includes two rows of results for the baseline method "BASE." The row actually labeled BASE in the first column is for the case when we normalize the residual errors in order to speed up the calculation of the running median from one local window to another in the baseline method. The second baseline-method row, labeled "BASE′" is for the case when no such normalization is carried out. Therefore, the second row of numbers is produced with a slower implementation of the baseline method. *As the reader can see, our normalization of the residual errors in order to*

TABLE 5
Average Angular Errors and Variances for Boxes Sequence

Method	Ave. angular error / variance without noise	Ave. angular error / variance with different variances of Gaussian noise			Ave. angular error / variance with different densities of salt-and-pepper noise		
		4	8	12	0.005	0.01	0.02
RVL-L1(5)	6.02/ 8.24	6.42/ 8.35	6.83/ 8.64	7.17/ 8.87	9.52/10.65	11.63/11.66	14.53/13.33
RVL-L1(25)	6.04/ 8.25	6.47/ 8.36	6.91/ 8.70	7.28/ 8.96	9.30/10.58	11.45/11.66	14.62/13.46
RVL-L2(5)	4.50/ 6.31	5.46/ 7.48	6.32/ 8.66	6.94/ 9.21	4.83/ 7.00	5.19/ 7.66	7.05/10.05
RVL-L2(25)	4.33/ 5.92	5.35/ 7.40	6.40/ 8.93	7.14/ 9.82	4.37/ 6.10	4.46/ 6.20	4.81/ 6.78
BASE	4.30/ 5.85	5.35/ 7.47	6.43/ 9.01	7.16/ 9.88	4.31/ 5.95	4.38/ 6.10	4.52/ 6.16
BASE′	4.33/ 5.95	5.36/ 7.46	6.43/ 9.02	7.17/ 9.89	4.31/ 5.94	4.39/ 6.09	4.53/ 6.18
NY	6.00/ 8.28	6.38/ 8.38	6.76/ 8.64	7.09/ 8.86	10.06/10.96	12.36/12.04	15.32/13.84

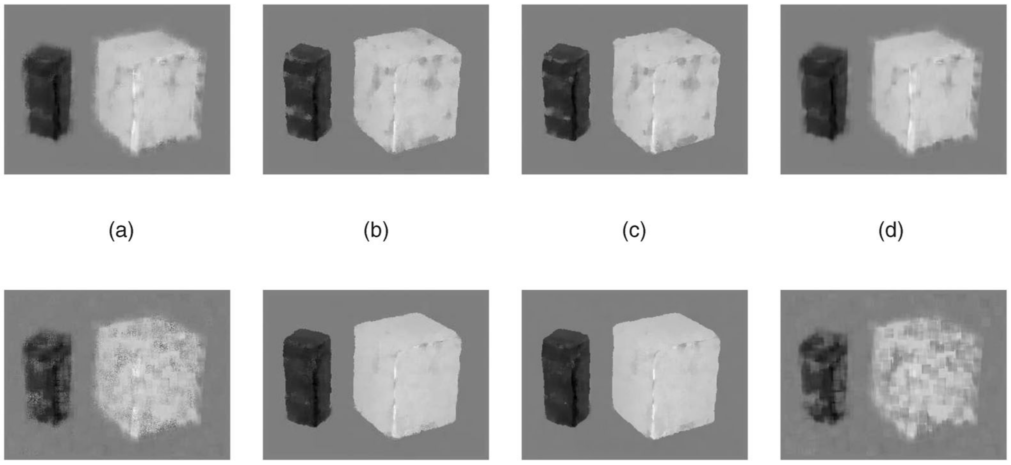


(a)



(b)

Fig. 15. Average angular errors for Boxes sequence with (a) Gaussian noise and (b) salt-and-pepper noise.



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

(e) (f) (g) (h)

Fig. 16. Optical flow results of (a), (e) "RVL-L1" method based on the standard LMedS, (b), (f) "RVL-L2" method based on the modified LMedS, (c), (g) baseline implementation, and (d), (h) "NY" method based on the LS for Boxes sequence perturbed by Gaussian noise with variances of 4, and salt-and-pepper noise with density of 0.005, respectively.

achieve a higher speed does not degrade the accuracy of the baseline method. For all of the motion estimation methods used in the comparisons shown in Table 5 and Fig. 15, we used 13×13 local windows and 5×5 subwindows.

As we see in Table 5 and Fig. 15a, the average angular error for the modified LMedS method is much smaller than that produced by the standard LMedS method at low levels of noise. This superior accuracy of the modified LMedS method can be verified visually at the motion boundaries in Figs. 16a, 16b, 16c, and 16d. (The darker color in this figure represent motions to the left, brighter colors motion to the right, and the gray color zero motion.) This tells us that the temporary solutions of the modified LMedS method, each obtained by a least-squares solution from all the pixels in a subwindow, are much more reliable than those of the standard LMedS method in which a temporary solution is obtained directly from the constraint equations using only as many pixels as the number of unknowns to be solved for.

An interesting difference between the performance of the standard LMedS and the modified LMedS takes place as the level of noise increases. By its very nature, the standard LMedS seeks that solution that is in the "middle" of the intersections of the constraint lines corresponding to all those pixels that are considered to be inliers. On the other hand, the modified LMedS method seeks that solution which corresponds to the dominant motion in the local window. But, when noise is high, it may not be easy to discern the dominant motion. This effect is visible in Fig. 15a and Figs. 16a, 16b, 16c, and 16d, where we see that the modified LMedS outperforms the standard LMedS when the image noise is moderate. However, as the level of noise increases, there is a reversal in the performance of the two.

Fig. 15a also shows the average angular errors of the two LMedS methods with different number of samplings. In this figure, we can see there is no significant difference between the two. From this, we can say that if noise affects all the image pixels randomly then selecting good temporary solutions does not much depend on the number of samplings. For the same reason, Fig. 15a shows no significant difference between the average angular errors for the baseline implementation and that of the modified LMedS method.

With regard to the results shown for the salt-and-pepper noise in Table 5 and Fig. 15b, the superiority of the modified LMedS method over the standard LMedS method is now much more evident than was the case with Gaussian noise. The average angular error for the modified LMedS method is now much smaller than that for the standard LMedS method at all noise levels. Figs. 16e, 16f, 16g, and 16h visually show the superior accuracy of the modified LMedS method, especially so at the motion boundaries. From this, we can conclude that the temporary solutions of the modified LMedS method are even more robust to salt-and-pepper noise than they are to

Gaussian noise. This can be explained as follows: Since the salt-and-pepper noise does not affect all the pixels in the image, the probability of having a good temporary solution from the randomly sampled set of pixels is now higher. However, as the number of pixels affected by the noise increases, the temporary solutions solved by the modified LMedS method are also affected and the average angular error of this method increases faster than that of the baseline implementation as shown in Fig. 15b.

Again, with reference to the results on the real data shown in Table 5 and Fig. 15b, we will now comment on the performance of the different methods with regard to the number of samplings for the case of salt-and-pepper noise. As shown in Fig. 15b, with this type of noise, the performance of the modified LMedS method deteriorates faster with increasing noise when the number of samplings is five vis-vis when it is 25. On the other hand, the standard LMedS shows substantially the same performance at both sampling rates. This shows us again that the temporary solutions of the modified LMedS method are much more reliable than those of the standard LMedS method and, therefore, the probability of having a good temporary solution of the modified LMedS method increases as the number of samplings increases. For the same reason, the average angular error of the baseline implementation is smaller than that of the modified LMedS method at large levels of noise.

6 Concluding Remarks

Our comparative study demonstrates the general superiority of the local integrated methods based on the modified LMedS regression technique over the other global or local methods. When comparing the LMedS-based approaches, the superiority of the modified LMedS-based method is particularly noticeably at low and intermediate levels of noise for the case of Gaussian noise and at all levels of noise for the case of salt-and-pepper noise. This tells us that the temporary solutions of the modified LMedS method are much more reliable than those of the standard LMedS method. The standard LMedS method was based on the standard LMedS regression as proposed by Rousseeuw and Leroy in [24].

We demonstrated this claim with the help of four ground-truthed data sets, synthetic and real, containing illumination variations that are artificial and real. One of our evaluation data sets was a nine-frame ground-truthed image sequence generated with OpenGL. Our real data set consisted of a two-frame ground-truthed sequence with real illumination variations.

In keeping with "best practice" in performance evaluation in computer vision, our error evaluation study incorporated two different types of noise, additive Gaussian, and salt-and-pepper. And, in keeping with the approaches used by others for analyzing the accuracy of motion estimation methods, our quantitative evaluation calculated the average angular errors associated with the estimated motion vectors. The error was calculated as a function of the noise variance for the Gaussian case and as a function of the drop-out rate for the salt-and-pepper case.

Our contribution in this paper also includes a gold-standard implementation of the LMedS-based motion estimation that others should find useful for studying the performance of their own LMedS-based approaches. The gold-standard implementation is not meant for real-time motion estimation by any stretch of imagination; its sole aim is to provide an implementation that uses all the data for forming temporary solutions and that therefore is guaranteed to have no errors on account of the random sampling effects. The gold-standard implementation of Section 4 is only for carrying out studies in the error performance of LMedS-based approaches.

Acknowledgments

The authors would like to thank Kwangtaek Kim for helping to generate the video sequence using OpenGL. This work was partially supported by the US National Science Foundation under grant 99-05848.

References

[1] P. Stumpf, "Über die Abhängigkeit der Visuellen Bewegungsrichtung und Negativen Nachbildes von den Reizvorgangen auf der Netzhaut," *Zeitschrift für Psychologie,* vol. 59, pp. 321-330, 1911.

[2] M.J. Black and P. Anandan, "A Framework for the Robust Estimation of Optical Flow," *Proc. Int'l Conf. Computer Vision,* pp. 231-236, May 1993.

[3] B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," *Artificial Intelligence,* vol. 17, pp. 185-203, 1981.

[4] S. Negahdaripour, "Revised Definition of Optical Flow: Integration of Radiometric and Geometric Cues for Dynamic Scene Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 9, pp. 961-979, Sept. 1998.

[5] B.D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereovision," *Proc. Int'l Joint Conf. Artificial Intelligence,* pp. 674-679, 1981.

[6] M.A. Gennert and S. Negahdaripour, "Relaxing the Brightness Constancy Assumption In Computing Optical Flow," MIT AI Lab. Memo, no. 975, 1987.

[7] H.W. Haussecker and D.J. Fleet, "Computing Optical Flow with Physical Models of Brightness Variation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 6, pp. 661-673, June 2001.

[8] N. Mukawa, "Optical-Model-Based Analysis of Consecutive Images," *Computer Vision and Image Understanding,* vol. 66, no. 1, pp. 25-32, 1997.

[9] A. Bab-Hadiashar and D. Suter, "Robust Optic Flow Computation," *Int'l J. Computer Vision,* vol. 29, no. 1, pp. 59-77, 1998.

[10] E.-P. Ong and M. Spann, "Robust Optical Flow Computation Based on Least-Median-of-Squares Regression," *Int'l J. Computer Vision,* vol. 31, no. 1, pp. 51-82, 1999.

[11] M.J. Black, D.J. Fleet, and Y. Jacoobs, "Robustly Estimating Changes in Image Appearance," *Computer Vision and Image Understanding,* vol. 78, no. 1, 2000.

[12] Y.-H. Kim, A.M. Martinez, and A.C. Kak, "A Local Approach for Robust Optical Flow Estimation under Varying Illumination," *Proc. British Machine Vision Conf.,* 2004.

[13] Y.-H. Kim, A.M. Martinez, and A.C. Kak, "Robust Motion Estimation under Varying Illumination," *Image and Vision Computing,* vol. 23, no. 4, pp. 365-375, 2005.

[14] S.-H. Lai and M. Fang, "Robust and Efficient Image Alignment with Spatially Varying Illumination Models," *Proc. Conf. Computer Vision and Pattern Recognition (CVPR),* 1999.

[15] J.-M. Odobez and P. Bouthemy, "Robust Multiresolution Estimation of Parametric Motion Models," *Visual Comm. and Image Representation,* pp. 348-365, 1995.

[16] D. Shulman and J.-Y. Herve, "Regularization of Discontinuous Flow Fields," *Proc. IEEE Workshop Visual Motion,* 1989.

[17] T.S. Huang, G.J. Yang, and G.Y. Tang, "A Fast Two-Dimensional Median Filtering Algorithm," *Acoustics, Speech, and Signal Processing Magazine,* vol. 27, no. 1, pp. 13-18, 1979.

[18] H.-H. Nagel, "On the Estimation of Optical Flow: Relations between Different Approaches and Some New Results," *AI Magazine,* vol. 33, pp. 299-324, 1987.

[19] S. Uras, F. Girosi, A. Verri, and V. Torre, "A Computational Approach to Motion Perception," *Biological Cybernetics,* vol. 60, pp. 79-97, 1988.

[20] N. Cornelius and T. Kanade, "Adapting Optical Flow to Measure Object Motion in Reflectance and X-Ray Image Sequences," *Proc. ACM SIGGRAPH/SIGART Interdisciplinary Workshop Motion: Representation and Perception,* Apr. 1983.

[21] N. Mukawa, "Motion Field Estimation for Shaded Scenes," *Proc. Int'l Conf. Image Processing (ICIP),* Sept. 1989.

[22] B.G. Schunck, "Image Flow: Fundamentals and Future Research," *Proc. Conf. Computer Vision and Pattern Recognition (CVPR),* June 1985.

[23] S. Negahdaripour and C. Yu, "A Generalized Brightness Change Model for Computing Optical Flow," *Proc. Int'l Conf. Computer Vision,* 1993.

[24] P.J. Rousseeuw and A.M. Leroy, *Robust Regression and Outlier Detection.* New York: Wiley, 1987.

[25] P.W. Holland and R.E. Welsch, "Robust Regression Using Iteratively Reweighted Least Squares," *Comm. in Statistics,* vol. A6, no. 9, pp. 813-827, 1977.

[26] S. Ayer, H.S. Sawhney, and M. Gorkani, "Model-Based 2D and 3D Dominant Motion Estimation for Mosaicing and Video Representation," *Proc. Int'l Conf. Computer Vision,* pp. 583-590, 1995.

[27] P. Nesi, A. Del Bimbo, and D. Ben-Tzvi, "A Robust Algorithm for Optical Flow Estimation," *Computer Vision and Image Understanding,* vol. 62, no. 1, pp. 59-68, 1995.

[28] B. Schunck, "Image Flow Segmentation and Estimation by Constraint Line Clustering," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 11, no. 10, Oct. 1989.

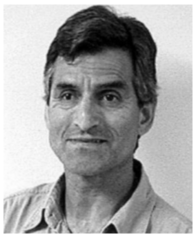
[29] R.M. Haralick, "Performance Characterization in Computer Vision," *Computer Vision Graphics Image Process: Image Understanding,* vol. 60, no. 2, pp. 245-249, 1994.

[30] M. Otte and H.H. Nagel, "Optical Flow Estimation: Advances and Comparisons," *Proc. European Conf. Computer Vision,* pp. 51-60, 1994.

[31] B. McCane, K. Novins, D. Crannitch, and B. Galvin, "On Benchmarking Optical Flow," *Computer Vision and Image Understanding,* vol. 84, pp. 126-143, 2001.

[32] J.L. Barron, D.J. Fleet, and S.S. Beauchemin, "Performance of Optical Flow Techniques," *Int'l J. Computer Vision,* 1994.

Yeon-Ho Kim received the BS degree from Seoul National University in electrical engineering, the MS degree from Korea Advanced Institute of Science and Technology (KAIST) in electrical and electronic engineering, both in Seoul, Republic of Korea, and the PhD degree from Purdue University in electrical and computer engineering, in West Lafayette, Indiana, in 1989, 1991, and 2005, respectively. From 1991 to 1998, he worked as a research staff in the Central R&D Center of LG Industrial Systems Co. (LGIS), in Anyang, Republic of Korea. At LGIS, he worked in kinematics, dynamics, and design of controller for industrial robots. From 2000 to 2005, he worked in the Robot Vision Laboratory at Purdue University. Currently, his research interests include computer vision, especially optical flow, robotics, and human-computer interaction.



Avinash C. Kak is a professor of electrical and computer engineering at Purdue University. He has coauthored the widely used book *Digital Picture Processing* (Academic Press, 1982). He is also a coauthor of *Principles of Computerized Tomographic Imaging*, which was recently chosen by SIAM Press for republication in their Classics in Applied Mathematics series. His latest book *Programming with Objects: A Comparative Presentation of C++ and Java* was published in 2003. His current research interests are focused on the sensory aspects of robotic intelligence, especially robotic vision. He is the chief editor of the journal *Computer Vision and Image Understanding* published by Elsevier.