m o n a s h

MECSE 96-4

Motion Estimation Using Robust Statistics

Alireza Bab-Hadiashar and David Suter

10 July 1996

TECHNICAL REPORT
DEPARTMENT OF ELECTRICAL AND COMPUTER SYSTEMS ENGINEERING,
FACULTY OF ENGINEERING,
MONASH UNIVERSITY, CLAYTON, 3168, VIC, AUSTRALIA.

Motion Estimation Using Robust Statistics

Alireza Bab-Hadiashar and David Suter
Dept. Elect. and Comp. Syst. Eng., Monash University
Clayton 3168, Vic., Australia
Email: ali.bab-hadiashar@eng.monash.edu.au

Abstract

A method for deriving optic flow, using robust statistics, is derived and applied to standard test image sequences. We compare the performance of our approach against the best results derived from competing methods. The method out-performs all published methods in terms of accuracy. The method is also cheaper to implement than most of the other methods. One of the key features in the success of this method, is that we use Least Median of Squares, which is known to be robust to outliers. However, the Least Median of Squares is known to be computationally expensive to compute. We manage to keep the computational cost very low, whilst still retaining excellent performance, by using an *approximate* solution to the Least Median of Squares *only in a first stage that detects outliers.* The approximate solution is fast and adequately robust, as our extensive experiments demonstrate. The essential features of our method should be applicable in the solution of a wide range of other problems: essentially, wherever one solves an over-determined set of linear equations.

1 Introduction

This paper describes a robust method for computing optic flow. Optic flow is the apparent motion of objects when projected onto the image plane. The crux of our method is to develop a robust way to solve a set of over-determined linear equations.

Despite at least two decades of research, the best methods for the extraction of optical flow are relatively inaccurate and non-robust. By non-robust, we mean that the accuracy, in particular parts of the image, is often considerably worse than the general accuracy attainable over much of the rest of the image. The degradation in accuracy is due to a number of factors such as larger noise in that region and/or failure of the underlying image motion model (see section 2.2).

In this paper, we use robust statistical methods, as well as some other innovations, in the motion recovery formulation. We are able to achieve both an increase in accuracy, and a degree of robustness that is matched, if at all, by few other methods.

The outline of the papers is as follows. In section 2 we give a brief introduction to motion estimation methods. In this section we also survey some of the methods of others, which also aim to improve the robustness of optic flow calculation. The purpose is not to give a comprehensive survey (that is beyond the scope of this paper) but to contrast our approach with that of the major competing methods. The motivation for our method comes from comparing the relative advantages and disadvantages of some standard formulations for solving a set of over-determined linear equations. In particular, we derive the basis for our approach by comparing standard least squares approaches with closest point and least median of squares re-formulations of the problem - section 3. In section 4 we describe our new method: we adapt some methods for fast (approximate) Least Median of Squares in Linear Regression problems, so as to be able to perform outlier detection for *(weighted) Least Squares* problems. We then, section 5, present results of experiments that verify the effectiveness of our proposed approach.

Since the key to our approach, is to solve an over-determined system of linear equations in a robust manner, we expect the approach will be more widely applicable to other areas (to other problems in computer vision and to other problems in unrelated areas that also reduce to the solution of an over-determined linear system). Thus, the present paper should be considered as the first demonstration of a basic robust scheme for solving linear equations, where we have chosen to apply the method to optic flow calculation. We do not claim that the current implementation is the best one can do with our basic method. Some suggestions for improving the implementation, for optic flow calculations; as well as suggestion of other areas of application of our techniques, are contained in our summary and conclusion (section 6).

2 Optic Flow and Robustness

It is possible to place almost all optic flow techniques into one of the following three categories:

- Correlation based techniques
- Phase or energy based techniques
- Differential techniques.

Correlation based techniques seek to explicitly match a small region of pixels in one image, with a region in the next image, using a measure of match that is related to cross-correlation. Such methods are similar to stereo or photogrammetric techniques. They are also popular in video coding methods that employ motion compensation (most MPEG encoders employ this type of technique).

Phase or energy based techniques seek to filter the image sequence with specially tuned filters - the outputs of which are combined to give a phase or energy measure that indicates the local speed and direction of motion. One of the most successful methods, but expensive to compute, is that of Fleet and Jepson [1].

Differential techniques try to relate local changes in image intensity (expressed as spatial and temporal derivatives of the image brightness function) to the optic flow. Differential techniques usually perform faster and usually lead to a simple set of linear equations. We base our approach in this class of approaches. For this reason, we explain, below, the basic idea behind this class of methods.

2.1 Differential Based Optic Flow Methods

The differential techniques invariably involve some form of what has become known as the "Optic Flow Constraint" (OFC). This constraint dates back to the late 1970's [2][3]. Although the derivation of this constraint occurs in many sources, we give a derivation here that is slightly more mathematically precise than most we have seen (see appendix A).

The OFC can be written as:

$$I_x u_x + I_y u_y + I_t = 0 \tag{1}$$

which relates the spatial (I_x and I_y) and temporal (I_t) derivatives of the image brightness function, at each point, to the optic flow, (u_x, u_y), at that point.

Since there is only one equation in two unknowns, we cannot solve for both the x and y components of the optic flow, without additional assumptions or information (the well known aperture problem). Put pictorially, (and in a way that we will make use of later), a single equation produced by the OFC only constrains the optic flow (u_x, u_y) to lie on a constraint line in 2D $u_x - u_y$ space: we need at least one other non-parallel constraint line to uniquely determine the flow.

In other words, using just the information we have so far, the problem is ill-posed. Various alternative strategies to make the problem well-posed (regularise the problem) have been suggested. These include: minimize a functional derived from the OFC and a smoothness penalty term (e.g., [3]), assume constant or affine variation in the optic flow (e.g., [4] [5]), and differentiate the OFC to obtain more than one constraint (e.g., [6]). All of these approaches, in some form or other, assume that the optic flow of nearby pixels are closely related - some form of motion consistency assumption. In the last example, differentiating the OFC, the nature of that motion consistency assumption is more subtle. Essentially, the "extra" equations are only independent because the derivatives have to be estimated by a process that collects information over a region (not a mathematical point) - if there is not some local consistency in the motion over this region, such a local gathering of information would not provide a valid constraint.

Regardless of the strategy for overcoming the aperture problem, one usually arrives at a set of linear equations to solve for the optic flow at each point:

$$Av = d \tag{2}$$

where v is a two component vector, $v = (u_x, u_y)$, (the optic flow we wish to derive), A is an $p \times 2$ matrix (whose coefficients are somehow related to the image brightness) and d is a p component vector (again, related to the image brightness). The rank of A should be greater than 2 (otherwise we still cannot solve for the two components of the flow). Note, although we only need two independent equations, we have to recognise that the derivatives can only be approximated, thus any two independent equations will not necessarily give the best estimate of the true optic flow. So we must, in general, use more equations than we have unknowns. In other words, we inevitably arrive at a set of over-determined linear equations.

We discuss the alternative general strategies for solving such an over-determined system in section 3.1. From the perspective provided by this discussion, we explain our approach, and the motivation for our approach in section 4.

2.2 Robust Optic Flow Methods

"Robustness" implies that the method is less sensitive to perturbations in input data. Roughly speaking, a robust approach should not produce estimates that are clearly wildly wrong. Nor should a robust approach, avoid making erroneous estimates simply through discarding potentially useful information, just to avoid the possibility of the result being erroneous: that is, a method that is overly conservative and produces no estimate almost anywhere in an image, is not a robust method.

2.2.1 Rationale for Robust Methods

Although we concentrate on optic flow methods based upon the differential optic flow constraints, much of what we say about the need for robust approaches still applies to the other classes of methods. In particular, in some form or other, all methods require some form of the following two basic assumptions:

- **Intensity Coherence** The image brightness of a point imaged in two successive images is (strictest assumption) constant or (weaker assumption) nearly constant.
- **Motion Coherence** The motion of points nearby in an image is the same (strictest assumption) or slowly varying (weaker assumption).

One common form of the last assumption is to allow the image velocity to locally vary in some linear or affine fashion.

There are a number of reasons why particular methods of optic flow produce erroneous or inaccurate results. It is useful to categorise these sources according to:

- failure of the image/motion model
  - failure of the brightness consistency (weak or strong forms)
  - failure of the motion consistency (weak or strong forms)
- noise (e.g., sensor noise, poor approximation of derivatives in a differential based scheme).

We argue that this classification is useful because failure of the image model, as we define it, has some important practical distinguishing features. Firstly, it usually occurs in particular regions that are associated with geometric or physical properties of the scene. For example, a moving specular surface may destroy the accuracy of the brightness constraint, the boundary between two regions moving with different motion will destroy the motion consistency assumption. Secondly, this class of errors tends to lead to more severe errors, and, in some sense, errors that are less random than the errors from the other sources that we have lumped together as "noise". For this reason, we argue that the first class of errors can be detected by robust statistical methods to detect outliers. The second class of errors, if also severe can be similarly detected as outliers. On the other hand, if errors from this class are less severe, and cannot be considered to be outliers, they can perhaps be reduced in effect by simple processes such as least squares solution. In this way we can take maximum advantage of the particular features of some standard robust and non-robust re-formulations of the optic flow problem - see section 3.1.

Thus, the essential point is that, we see robust methods as ones that can avoid producing wildly wrong estimates for the optic flow, *particularly where the source of the likely breakdown in performance is a failure of the image motion consistency or image brightness consistency assumptions.* Such breakdowns commonly occur, with non-robust methods, around the edges between differently moving objects. An extreme case of the latter is the situation where we have "motion transparency" (for example, a reflection on a transparent surface moving differently to the material beyond that surface; or, where there are two populations of movement interspersed, such as a flock of birds against a background of clouds) - an example of which is shown in section 2.2.2.

2.2.2 An Example of Motion Transparency and Outliers

To demonstrate the effects of noise and of breakdowns in the underlying motion model, we take a real image sequence: the famous Hamburg Taxi scene (see figure 1). A van in the lower left corner is moving to the left at approximately 3 pixels per frame. We selected a small window, centred on this van (13 pixels square and centred at the point (220,130)). For each pixel, we calculated the spatial and temporal derivatives of the image brightness to yield a single OFC for each pixel. We then plotted the OFC lines (see figure 2). If, in this small rectangle, we had a single underlying image motion, and the data were noiseless, we would expect all constraint lines to pass through the point representing that motion, (-3,0) in this example. However, even though there is some tendency for many of the lines (shown solid) to cluster about this point, there are many lines (shown dashed) that nearly intersect at the point (0,0). This is mainly due to the branches of the tree that intersect the rectangle from which we have drawn our pixels (the pixels in this rectangle, therefore, are a mixture of those belonging to the van, and those belonging to the tree: the former have approximately uniform motion to the left and the latter are stationary). This is a phenomenum sometimes referred to as "transparent motion" since it is similar to the situation where a reflection, in a pane of glass, for example, has different motion to other objects behind the reflection. In addition to the breakdown in the motion consistency or coherence assumption (i.e., the motion model) we have other noise effects, due to a variety of other sources, that cause a spread in the intersections of these two populations.

The point we wish to make, with this example, is that a non-robust method that tries to treat all of the constraint lines as being valid data (for a single underlying motion), will produce very poor estimates of the velocity. We wish to have a robust method that will reject the influence of any data that either belongs to another population (two or more independently moving object in a window), or is so badly corrupted by noise as to be unreliable. After briefly surveying other attempts to achieve this end (next section), we devise such a robust method (section 4).

2.2.3 Previous Robust Approaches

The explicit usage of robust statistics for recovering the visual motion dates back to early this decade. Darrel and Pentland [7] considered using M-Estimators for 3-D translations with constant depth. For (2-D) optic flow, Black and Anandan [8] and Odobez and Bouthemy [9] developed methods using M-Estimators for a correlation and differential formulation of the optic flow problem. The main difference between these two approaches is the use of different minimisation techniques. However, M-estimators have very low "break down points" [10] - the percentage of contaminated data that can cause the estimator to give an estimate far from the true estimate. In fact, the breakdown point of the proposed M-estimators is at most $\frac{1}{(p+1)}$ where p is the number of estimated parameters [11]. This shows that for an affine model used in [9], with 6 parameters, the breakdown point is only 14%. A second limitation of this

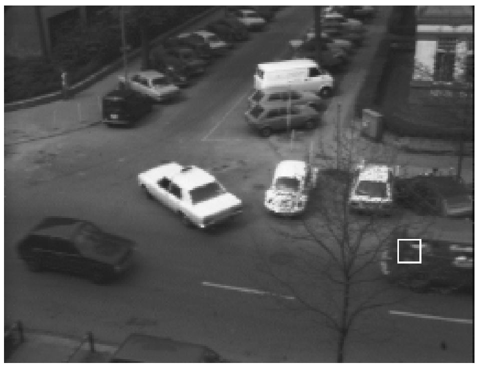

Figure 1: Hamburg Taxi Sequence

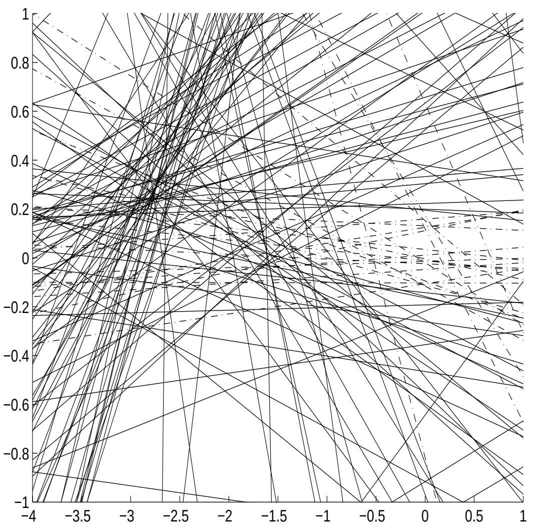

Figure 2: Hamburg Taxi Sequence - Optical Flow Constraints
Optical Flow Constraint Lines for the Hamburg Taxi sequence. There are 169 lines, one for each pixel in the rectangle in figure 1. The lower axis represents the horizontal velocity, u_x, and the left-hand axis represents the vertical velocity u_y.

approach is that there are a number of parameters to "tune" with no clear rule as to how to do so.

A similar approach for estimating the flow field, using the Horn and Schunk [3] type regularisation technique has also been proposed by Iu [12]. In this method, the motion is estimated by minimising the energy function of a globally smooth motion model. A rank ordering technique is used to reject the local outliers of the globally smooth model and the resulting minimisation problem is solved using a stochastical optimisation technique (simulated annealing). Aside from computational burden of such minimisation technique, the proposed kind of outlier rejection is likely to become very unstable for centrally located data. Since there is no measure on quantitative performance is presented, it is difficult to assess the performance of this method.

In order to cope specifically with motion boundaries, Fennema and Thompson [2] proposed a clustering method using the Hough transform. This method is computationally very expensive. Schunck [13] modified this method, to reduce the computational cost, by clustering constraint lines along the OFC produced by the central pixel in a patch. However, such a method is patently non-robust to perturbations in the data that determined the central constraint line. To overcome that problem, Nesi et al. [14] proposed a method based on the Combinatorial Hough Transform. This approach, however, still remains expensive. Moreover, all of the methods just discussed, solve the problem as a version of the Closest Point Problem (see section 3.1.2, where we argue that this unnecessarily discards some useful information and, even worse, such a formulation can be highly influenced by less accurate information).

Another approach to optic flow computation using robust statistics is the use of Robust Hough Transform [5][15]. In this approach, an affine model of motion is introduced and the flow is calculated by Hough Transform. This Hough transform is based on the parameter space of the affine motion model, and "voting" uses the error between the brightness values of corresponding (using the affine mapping) pixels in sequential frames, weighted by a kernel function belonging to the M-Estimators family. The Median of Absolute Deviation is used to scale the residual before applying the kernel function. Since a good match should lead to a low error function (and thus a low vote in Hough transform space), the problem then reduces to an optimization problem: finding the minimum in Hough space. The resulting optimisation problem is then solved using the "steepest descent" method. This algorithm has serious limitations due to the robustness limitation of M-Estimators (see previous discussion of Black's method). Moreover, for the optimisation scheme to work, the support function has to be a well-behaved function. As explained by the authors, the function is only well-behaved in the region with valid Taylor expansion. So, in the regions with motion boundaries, transparency etc, the optimisation scheme is likely to fail. The very poor results presented for the Yosemite Image sequence [15] confirms our intuition about the limitation of their algorithm.

Several works use Total Least Squares (TLS) [16] [17] [18]. Even though such methods can improve the results (basically by considering noise in both the coefficients and the right hand side of equations such as 1) these methods *are not robust methods* as the break down point occurs even at one bad data sample.

Although Meer et al. [19] uses LMedS in other problems drawn from the computer vision field, it seems that no previous work has used Least Median of Squares (LMedS) to solve the optic flow problem. Mitiche [20] has mentioned the possibility of using LMedS, but appears not to have explored this suggestion with an algorithm or experimental results. We base our new method, in part, on the LMedS technique. However, as we describe later, the LMedS is not a practical technique by itself: thus our full technique is vastly different to even anything that could result directly from the suggestion of Mitiche.

We have not given a precise definition of "a robust approach", so it is not possible to give a complete categorisation of previous approaches into robust or non-robust. It is certainly true that most recent methods employ some form of post-processing (to identify and reject erroneous results). Although this produces a form of robustness, this is a form that, even if it works well, is too conservative - it often produces no estimate even where the situation ought to be salvageable (in short, the perturbed data has already "done the damage" and should have been removed earlier).

3 Methods for Solving Linear Equations in Presence of Noise

Before describing our approach in detail, we need to introduce various abstract problem formulations: Least Squares Problem, Standard (linear) Regression Problem, Least Median of Squares, and Closest Point Problem. We remind the reader of the context (refer to section 2.1): our brightness consistency and motion consistency assumptions generate, for each local area, a system of simultaneous equations (equation 2). There are a number ways of viewing and re-formulating the problem; many of them suggested by considering that each equation can be viewed as a constraint line in $u_x - u_y$ space. It is in this context, that our optic flow problem can be related the aforementioned abstract problems - which we now discuss, in turn.

3.1 Standard Reformulations of the Solution of Linear Equations

3.1.1 Least Squares Problem

This is the most well known formulation applicable to solving an over-determined set of equations.

Suppose we have p equations in 2 unknowns (x and y):

$$a_{i1}x + a_{i2}y = d_i,\ d = 1 \dots p \tag{3}$$

We seek the (x, y) that minimises:

$$E_{LS} = \sum_{i=1}^{p} (a_{i1}x + a_{i2}y - d_i)^2 \tag{4}$$

This formulation has a well known explicit solution (simply obtained by differentiating E_{LS}, with respect to the unknowns, and equating these derivatives to zero):

$$v = (A^T A)^{-1} A^T d \tag{5}$$

where $v = (x, y)$, A is the matrix with ij^{th} entry a_{ij} and d is a (column) vector with row d_i. Thus the formulation leads to easy and quick solution methods.

Another attractive feature of the least squares problem is that the LS produces an estimate that has the smallest variance amongst the solutions that are linear in the data d *when there are no systematic errors* [21]. The emphasis on the last qualifying phrase is placed there as this is the key weakness of the LS approach in optic flow calculations; particularly in regions, such as occluding boundaries, where the underlying model is no longer valid. This observation is one of the foundations of our proposed robust method.

3.1.2 The Closest Point and Standard Regression Problems

Many formulations of optic flow use a closest point problem (see section 2.2.3). Given a set of p lines: $y = m_i x + n_i,\ i = 1 \dots p$, the problem is to find the point that has the minimum sum of squared vertical distances to the lines. That is, to find the (x, y) that minimizes:

$$E_{CP} = \sum_{i=1}^{p} (m_i x + n_i - y)^2 \tag{6}$$

On the face of it, this problem looks very much like the Least Squares Problem. However, there is one very important distinction: in the Least Squares Problem, even though every equation represents a line, we have not chosen to weight each line equally. Put more precisely, we can map one problem on to the other by the following process: multiply each term in equation 4 by $\frac{-1}{a_{i2}}$ and identify $\frac{-a_{i1}}{a_{i2}}$ with m_i, and $\frac{d_i}{a_{i2}}$ with n_i. However, the scaling, which we have introduced to show the equivalence, will weight each term in the sum differently: some "errors" will now be more highly penalized by others in the reformulation. This distinction is important since our equations often carry a natural scale. For example, the OFC, equation 1, produces coefficients that are scaled by the gradient of the image brightness. This has the natural and useful consequence that those constraints resulting from parts of the image with high contrast will carry more weight, in a least squares formulation (an observation commonly made in the literature - see, for example, [22]). To arbitrarily rescale such natural weights, as one would have to do to reformulate the problem as an instance of the closest point problem, is likely to be a retrograde step - an objection to this type of approach that seems to have escaped many (e.g, [14]) who have used a closet point formulation in their schemes.

There is a one-to-one relationship between the closest point problem formulation and with that of the standard regression problem (and hence with least squares formulation). In the standard (linear) regression problem, we have a number of data points, to which we wish to fit a line. That is, given a set of p points: $\{(x_i, y_i) : i = 1 \ldots p\}$, we wish to find the line, parameterised by (m, n), i.e., $y = mx + n$, such that the sum of squared residuals (squared vertical distances to the line from the points) is minimised. In other words, we seek (m, n) that minimise:

$$E_{SR} = \sum_{i=1}^{p} (mx_i + n - y_i)^2 \tag{7}$$

Given a CP formulation, we can define a mapping T that takes each line (m_i, n_i) of a CP formulation and uses these parameters to replace (x_i, y_i) of a SR formulation. Comparison of formula 6 and formula 7 shows that the solutions will be related by: the original CP solution (x, y) is given by $(m, -n)$, where (m, n) is the solution of the new SR formulation. Obviously, a similar relationship applies in reverse.

One of the problems with the standard regression formulation, with the least squared formulation, and and with the closest point formulation, is that one bad data point can drastically influence the solution. These methods have a break down point of one and are not robust.

3.1.3 Least Median of Squares (LMedS)

The LMedS problem, as proposed in [23] [11] is a reformulation of the standard regression problem. Instead of finding the line that has the smallest squared vertical distances from the data points, the LMedS approach identifies the narrowest strip (bounded by two parallel lines) that contains one more than half of the data points: the LMedS line then runs down the middle of this strip [24]. The break down point of the LMedS is 50% - it can tolerate up to half of the data points being arbitrarily bad! As long as the majority ($\frac{p}{2}$+1, in our examples) are "sensible", in some sense, the solution will be "sensible".

As we described before, our preferred formulation for the optic flow problem is a Least Squares formulation. The LMedS method presented in [23] is a robust solution to the Standard Regression problem: therefore, we have reformulated the LMedS to apply to the Least Squares problem (see section 3.2). In the sequel, when we refer to the LMedS approach, we refer to the approach as modified for our purposes.

We now illustrate how the LMedS solution can provide a more robust solution, than LS. We generated 81 linear equations, chosen so that 43 of the associated lines pass through the point $(3, 2)$ and the remainder of lines pass through $(-1, -2)$. This can represent a situation where we have two differently moving objects in one patch of pixels, the objects are positioned so that we obtain approximately equal mixture of data consistent with one motion $(u_x, u_y) = (3, 2)$ (this is 53.1% of the lines) and the rest consistent with a motion of $(-1, -2)$. We then imposed 50% random noise to the coefficients of the lines associated with the (narrow) majority. The coefficients of the rest of the lines are untouched. One can see the lines plotted in figure 3. This represents an extreme case. Not only are nearly 50% of the data consistent with some other estimate, but this slight minority data is all "clean" - they are all perfectly in agreement with the solution $(-1, -2)$. The slight majority population has been polluted with noise so that there is no perfect agreement, of this slight majority, with the motion $(3, 2)$. However, despite the apparent difficulty of finding the "true" majority solution, the LMedS does, indeed, predict virtually the correct result (the solution by this method is $(1.909, 3.071)$, whilst a Least Squares solution is, as expected, very inaccurate.) Indeed, the Least Squares solution of $(0.686, 0.558)$ is, as expected, consistent with neither population of data but is consistent, wrongly, with some sort of compromise.

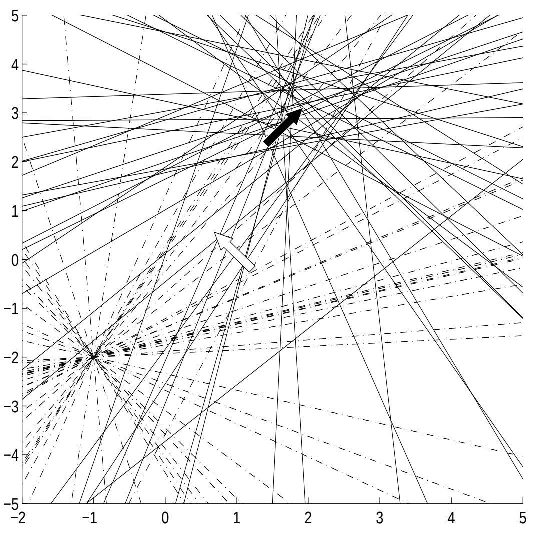

Figure 3: LMedS Example
In this figure, there are 81 lines. Of these, a slender majority (43) are formed from lines that were coincident on $(2, 3)$, however, they have been displaced by 50% noise having been added to their coefficients; and the rest are coincident on $(-1, -2)$. The LMedS solution (solid arrow) is still in agreement with the intersection point of the majority (before noise), whilst the LS solution (outlined arrow) is some type of erroneous compromise between the two populations. This shows the extreme robustness of the LMedS formulation.

3.1.4 Summary and Comparison of the Formulations

The CP and SR problems are essentially dual formulations of the same problem. The LS formulation is related to the CP and SR formulations but not an identical reformulation. The LS, CP, SR, are all fast to compute; however, they are all non-robust. The LMedS is very robust; but is expensive to compute. Indeed, no closed form solution of the LMedS formulation is known. The fastest known method for computing the exact LMedS solution has $O(p^2)$ time and $O(p)$ space complexity [25].

3.2 LMedS for Outlier Detection - WLS

The previous discussion has shown that, if we try to formulate the solution of the over-determined set of linear equations (our optic flow constraints) into any of the LS, CP, or SR formulations, we may gain such things as fast solution algorithms or an improvement in the ease of analysis of the formulation, *but they are all basically non-robust.* The LMedS is robust but computationally and analytically unattractive. The key to salvaging the situation, in view of this dilemma, has two parts, explained in this section. Firstly, there are fast methods to *approximately* solve the LMedS solution. The approximation is usually good enough to employ the LMedS solution, not as a final solution to our optic flow problem, but as a temporary solution from which we can detect outliers. Secondly, having detected outliers, we can now safely use a fast, but non-robust method, such as LS, on the data after the outliers have been removed.

Using this observation, we will (next section) devise a complete method for optic flow calculation that has, as a basis, what we call Weighted Least Squares. The essential idea is that, after solving, approximately, the LMedS formulation; we then have an estimate of the dominant optic flow (in that region), from which we can classify the constraint lines as being "good" or "bad" in terms of how consistent they are with this estimate. We can then "weight" the constraint equations (associated with each line) to reflect this confidence measure. In the extreme case (one that we use in this paper) the weights are either 0 (reject) or 1 (accept). We can then solve the weighted equations by a non-robust method, such as least squares, since we have, hopefully, removed all significant outliers.

The complete scheme is a little more complex than this. However, at this this point, before

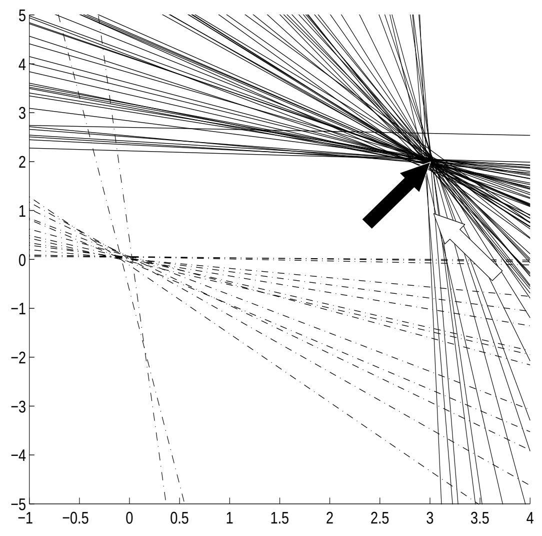

Figure 4: Weighted Least Squares
In this diagram there are two populations of lines - the majority (solid) are consistent with one common point, while the minority are consistent with another. The Weighted Least Squares solution (black arrow) is unperturbed by the presence of the minority population (outliers) whilst the Least Squares solution (white arrow) is consistent with neither population.

explaining the complete scheme in detail, we take the opportunity to illustrate the effectiveness of the key idea we have just introduced.

First, consider a synthetic example. We define the parameters of 81 lines so that the majority (all except 16) pass through the point (3,2) while the minority pas through the origin. These lines are drawn in figure 4. The black arrow depicts the (robust) weighted least squares solution (3.006,1.992), which is much closer to the true intersection of the majority than the least squares solution of (3.037,0.938).

For our second example, we return to our Hamburg Taxi example: figures 1 and 2. The least squares (LS) and weighted least squares (WLS) solutions are depicted in figure 5. The WLS solution of $(-2.703, 0.109)$ is clearly closer to the true motion (not precisely known but is approximately $(3,0)$) than the LS solution of $(-2.262, 0.211)$.

4 Proposed Method

We now have all the ingredients (and background rationale) for constructing our method. Given a set of linear optic flow constraint equations (equation 2), we would, ideally have a single unique solution. Pictorially, in $u_x - u_y$ space, each equation is a line and, in the ideal case, all lines would go through a single point - our required solution. However, in reality, we are faced not only with a situation in which most lines do not go through the real solution point, but that there are severe outlier lines. We are then faced with the question as to how to characterise the solution. Many alternatives suggest themselves: is it the point with the smallest sum of squared distances (taken vertically to the lines)? However, this is non-robust to our outliers. Moreover, as we argued earlier, the implicit normalisation of the CP formulation destroys the natural confidence weighting, available in the LS formulation, that comes with the magnitude of the coefficients of the linear constraints.

We would like to use the LS method for solving our optic flow equations (equation 2) because it is fast, the solution is explicitly known (equation 5), and the method is reasonably tolerant of *non-systematic* noise. We consider using LMedS, as an alternative, since it is robust, but we find that it is too expensive to use this method. Thus we return to consideration of LS. Those systematic errors, and/or other noise that is large enough to be troublesome, must be removed as the LS is not robust against these. At this point, we are saved because, although the LMedS is expensive, there are fast approximations will yield a solution good

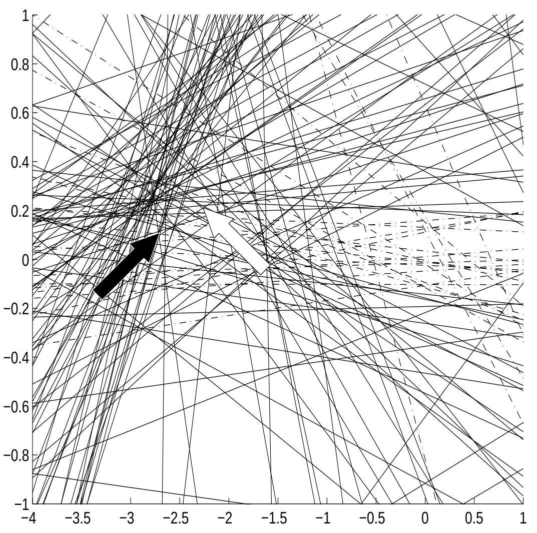

Figure 5: Weighted Least Squares - Hamburg Taxi
The black arrow shows the WLS solution and the white arrow shows the LS solution. The WLS solution is clearly closer to the (approximately) known true motion of the taxi (-3,0).

enough (close enough to the true solution) *to classify and remove outliers.* Thus we use the approximate solution to detect outliers, then, with the outliers removed we solve the remaining system with a least squares which is fast and gives a good estimate if the noise is non-biased.

To this basic strategy we need to add one further ingredient. The whole basis of the Least Median of Squares method is that there must be a population that is in the majority. This can, of course, break down (if we have, say, three populations roughly equal in size). Moreover, since we are using an approximate LMedS solution, and because there may be other disturbing factors (perhaps there simply is not enough texture, in the region to give any reasonable constraints); we need to be able to validate our final answer (and reject estimates that are, despite our best efforts according to the previously outlined strategy, clearly erroneous). We shall propose a measure of reliability (appendix C) for this purpose. Thus our scheme requires just two thresholds: a threshold for outlier detection (appendix B) before applying WLS, and a threshold for reliability to discard any final answers, after WLS, that are still unlikely to be accurate. Since the precise method for choosing these thresholds is not essential to the scheme we are proposing, we give the details in the appendices. Suffice to say, one threshold is always set (appendix B) to a fixed value, determined experimentally; and the other is set by a user defined value.

For purposes of clarity, we list the steps of our algorithm here:

1. Estimate the spatio-temporal derivatives of the image brightness function. The precise form of estimate, whilst important for accuracy, is not essential to our proposal. We choose to use, for our experiments, convolutions with derivatives of Gaussian functions (as is customary in many approaches, see for example [26]).

2. Select a patch of the image, over which we are going to assume some motion consistency. The precise form of the motion consistency is not essential: we are simply assuming a single or dominant population (we only recover the dominant population if there is more than one - we can, of course, elaborate our method to remove the dominant population and re-solve for any secondary populations, but we leave this to future work). In our experiments we choose to assume the motion is (spatially) constant in a patch. Future work could include an affine (spatial) variation in motion within a patch.

   As a result of this stage, each patch yields a system of equations expressing the motion constraints (equation 2), where, in our experiments, each constraint is of the form of equation 1.

3. Use a fast and approximate LMedS solution to obtain a temporary estimate of the

motion. Again, the precise method to approximate the LMedS solution can vary. We use here an algorithm adapted from Rousseeuw and Leroy [11]. Whereas that method was defined for a Linear Regression problem, we adapt it to a Least Squares problem.

One simply randomly chooses some fraction of the constraint lines, pairwise, and for each pair, we calculate the intersection. For this intersection one can calculate the residuals for all other lines, and, in turn, calculate the median of these residuals. The pair of lines with the smallest residual is chosen as the pair that defines the approximate Least Median of Squares solution. In a naive approach to approximating the LMedS solution, one would try to use every possible pair of lines (a combinatorially explosive situation) and evaluate the median of the residuals produced by the intersection of each pair. However, one can use a very small fraction of the possible pair combinations and the probability is high that the subsample will produce an intersection that belongs to the majority population [11] (for approximate LMedS in outlier detection, we need only *one* such good intersection). Indeed, if one chooses m pairs of lines, from the p we have in each patch (p is considerably larger than the number of parameters to be estimated), then the probability of this sample giving a good estimate for the LMedS solution is:

$$1 - (1 - (1 - \epsilon)^2)^m \tag{8}$$

where ϵ is the fraction of samples that do not belong to the majority population. From this formula, one can see that one can choose a very small population m, and still such that the probability is close to 1. We often use m as small as 10-30.

4. Reject outliers using a method of outlier rejection, based upon the temporary estimate of motion.

5. Solve the Weighted Least Squares problem resulting from the previous step. In our experiments, since we use weights 0 (reject) or 1 (accept), this is simply a matter of removing the rejected equations from equation 2 and solving by least squares, the smaller system, according to equation 5.

6. Examine the result, using a measure of reliability and do not produce any estimate if the result is judged to be still unreliable.

In our experiments, we repeat the above process for a patch centred upon every pixel, to yield the estimated of the flow at that pixel.

5 Experimental Validation

Before we detail the experimental results, we briefly discuss the computational cost. Properly comparing the computational cost of algorithms is, of course, a difficult procedure. Various optimizations can usually dramatically change the time it takes an algorithm to run; as can hardware implementation. Moreover, the speed can be affected by various parameter settings: in our approach, two significant settings are the size of the patch, and the number of pairs of lines we use to approximate the LMedS result in that patch. For these reasons, since our code (and usually the code of other researchers) is written more for correctness than speed, we give only rough, indicative times. Running on an SGI Indy (SC 4600 at 132MHz) our code takes approximately 6-7 minutes to calculate the flow for an image of size 252 by 316, using patches of size 7 by 7 around each pixel, and using 30 pairs of lines in each patch to approximate the LMedS. We can get good results faster, by using only 10 pairs of lines per patch, for example, and the running time is roughly halved. By comparison, a 6 level implementation of Black and Anandan's scheme [8] takes about 4 minutes, and Fleet and Jepson's scheme [1] takes about one hour or so. In other words, our scheme is much cheaper than Fleet and Jepson's and probably as cheap as Black and Anandan's (roughly). We see no impediment to making the scheme real time using hardware implementation at relatively modest cost, but, of course, proof of this is beyond the scope of this paper.

The quantitative performance of the proposed algorithm has been measured by applying the algorithm to image sequences for which the true flow fields are known. We provide quantitative performance figures for both synthetic and real image sequences. The synthetic image sequences contain a controlled number of the features, exhibited in real image sequences, that violate the basic model assumptions (motion consistency and image brightness consistency): thus they provide very optimistic (upper) bounds on the performance one can expect. However, synthetic image sequences do have two experimental advantages which make them useful: one can easily calculate the true motion field (indeed there are relatively few real sequences with known motion fields in general use in the vision community), and one can control the ways in which the basic model assumptions are violated. In our examples, the synthetic sequences basically violate just one model assumption: there is a motion discontinuity (and a simple and precisely known one at that) that allows us to investigate the performance of the algorithm in quantitatively identifying the failure of the simple motion model (and correctly reporting one of the two motions in a patch near the boundary).

All of the derivatives of the image brightness function are calculated by constructing the appropriate derivative of a 3D Gaussian function (with equal spatial and temporal standard deviation). Each sequence is then convolved with these derivative of Gaussian functions. We report the results for different sized spatial rectangular windows (within which the motion is assumed to be approximately uniform).

The error analysis is performed using Barron's [27] software. Therefore, the errors are reported in "degrees". This measure is the angle between the true and estimated motions when each is expressed in homogeneous coordinates. We refer to the cited paper for full details. As stated by Otte and Nagel [28], the values of the errors reported by this measure should be treated with some suspicion as estimates that have the same magnitude of error may provide vastly different angular errors. However, there is not a satisfactory summary statistic in use in the literature, and, since Barron's measure is becoming widely used, we believe that, providing the above warning is noted, it is one of the best available measures one can provide.

The error statistics are generally only of interest in some comparative sense - compared to other competing methods. Since Fleet and Jepson's method [1] is reasonably acknowledged as one of the more accurate methods (and one of the more expensive!) [27], an observation we have generally confirmed in our own experiments, we provide the results of applying this method to the same image sequences. The results for this method are generated by the software used in [27] (either by taking the results directly from those quoted in that paper, or, where such results were not provided, running the very same software ourselves). We have also, where available, quoted other results, from the literature, produced by other methods such as those of Szeliski and Coughlan [4].

5.1 New-Sinusoid1 Image Sequence

We created a sinusoidal image sequence similar to Sinusoid1 of [27]. The sequence contains images having the same spatial frequencies as Sinusoid1. However, in contrast to that sequence, which had spatially constant motion across the whole image, our sequence has motion boundaries. This was achieved by creating a stationary square (length of side 50 pixels) in the middle of each image. Figure 6 shows a sample image from the sequence and table 1 presents the error statistics for the method of Fleet and Jepson and that of our own. From this table, it is seen that our method clearly out-performs the Fleet and Jepson method in both accuracy and density of the points for which estimates are provided.

From figures 7, 8 and 9, we can clearly see that our validation procedure correctly removes the unreliable motion estimates (and, in this simple case, the unreliable motion estimates are those around the boundary of the central rectangle: where the image motion model breaks down). The size of the improvement can be judged by comparing successive rows in table 1 (WLS without check or validation compared with WLS using same size patch, etc., but with validation using $R^2 = 0.9999$). The validation procedure reduces the average error greatly but still retains a very high density of reported motion estimates.

5.2 Yosemite Image Sequence

The Yosemite sequence is one of the most complicated synthetic sequences that is widely used in the research community. The sequence was generated from digital terrain data of the Yosemite valley and the sequence depicts a simulated "fly-through". The motion is mainly

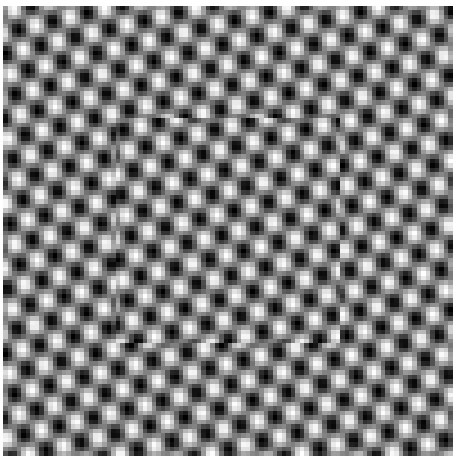

Figure 6: New-Sinusiod1 Sequence
One frame taken from the New-Sinusiod1 sequence. The texture is created by sinusoidal spatial variations.

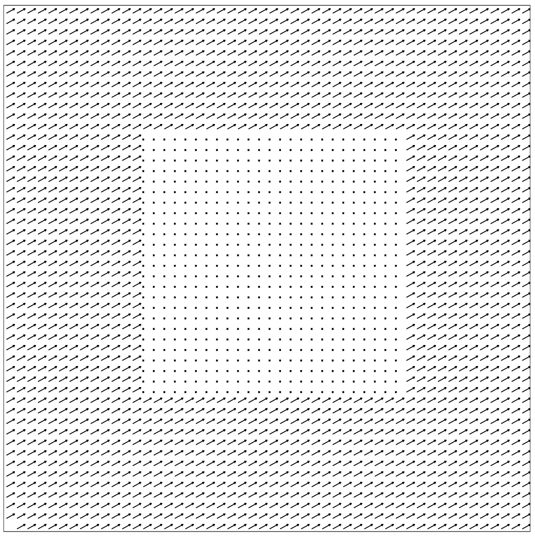

Figure 7: New-Sinusiod1 Sequence - Correct Flow
The correct motion for the New-Sinusoid1 sequence. Small $*$ symbols denote zero velocities at those positions.

Frame	Technique	Avg. Error	Std. Dev. Error	Density
9	Fleet and Jepson ($\sigma = 2.5$, $\tau = 1.25$)	7.39°	10.84°	43.4%
	Fleet and Jepson ($\sigma = 2.5$, $\tau = 2.5$)	1.41°	3.65°	46.0%
	WLS ($\sigma = 1.0$, 5×5, $m = 30$, without check)	2.15°	10.11°	100%
	WLS ($\sigma = 1.0$, 5×5, $m = 30$, $R^2 = 0.9999$)	0.05°	0.06°	83.9%
10	WLS ($\sigma = 1.0$, 5×5, $m = 30$, without check)	1.90°	8.59°	100%
	WLS ($\sigma = 1.0$, 5×5, $m = 30$, $R^2 = 0.9999$)	0.05°	0.06°	84.1%
11	WLS ($\sigma = 1.0$, 5×5, $m = 30$, without check)	1.41°	7.12°	100%
	WLS ($\sigma = 1.0$, 5×5, $m = 30$, $R^2 = 0.9999$)	0.05°	0.06°	84.6%

Table 1: Error analysis using New-Sinusoid1 image sequence
The second column of entries determines the method applied to generate the row of error statistics. In our method (WLS), the numbers in brackets depict the size of the Gaussian smoothing (σ is the standard deviation of the filter), the size of local patch used (p), the number of pairs of lines used to approximate the LMedS (m), and the reliability threshold (R^2), in that order.

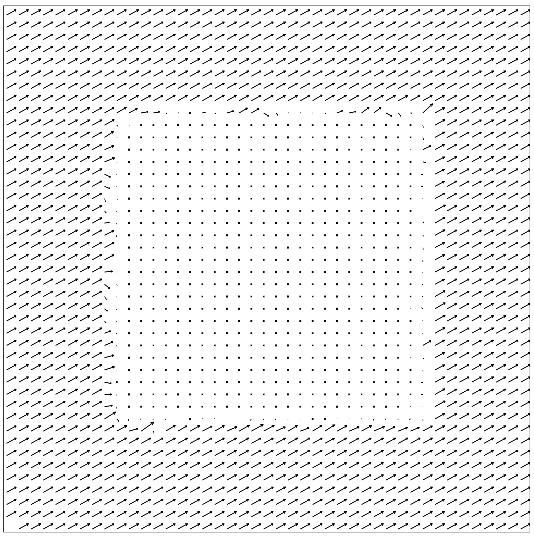

Figure 8: New-Sinusiod1 Sequence - Flow Calculated with WLS without validation
Without the validation procedure, to detect motion estimates that do not fit our image model well, we still have some erroneous estimates *along the boundaries of the stationary rectangle.*

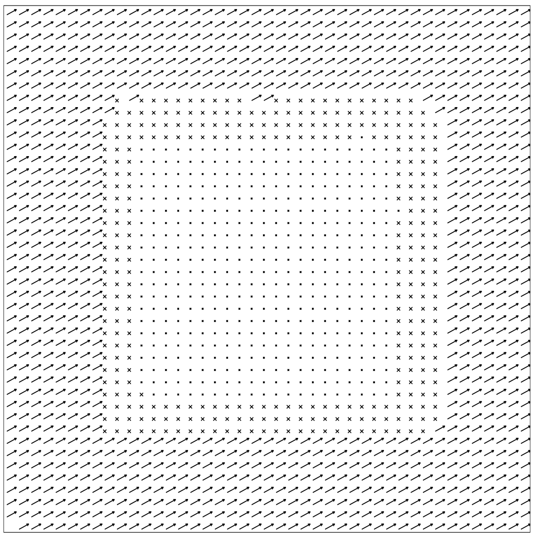

Figure 9: New-Sinusiod1 Sequence - Flow calculated by WLS with validation
Representative example of the result of applying the validation procedure ($R^2 = 0.9999$) to remove unreliable motion estimates. The positions marked with a $\times$ are those where the estimates were judged to be too unreliable.

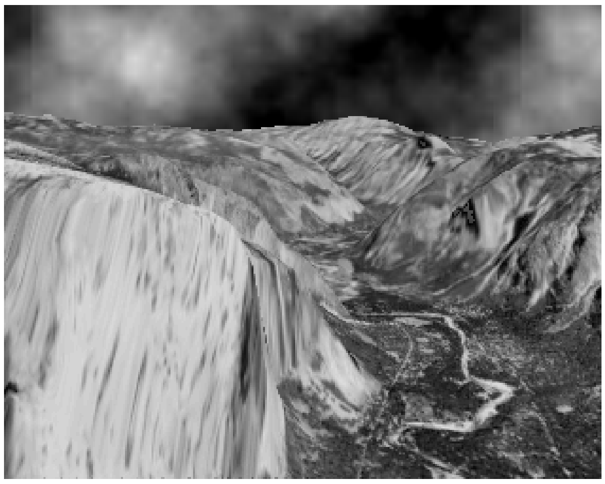

Figure 10: Yosemite Sequence - with cloud

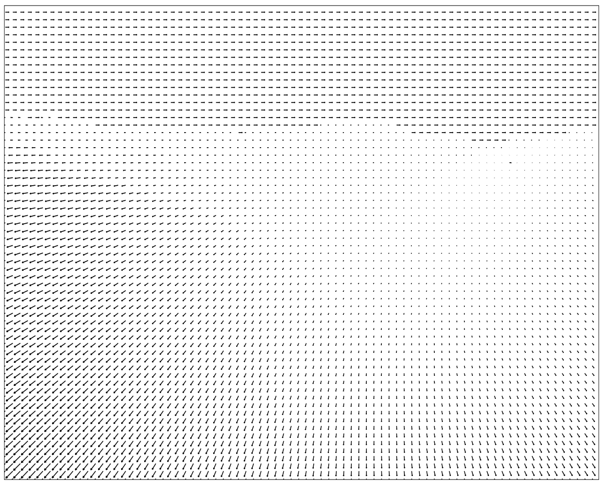

Figure 11: Yosemite Sequence Correct Flow - with cloud

divergent, while the clouds drift towards the right with a speed of 2 pixel/frame. The sequence is poorly sampled in time and the larger motions are, therefore, subject to bad temporal aliasing. The results of using this sequence in our experiments are shown in table 2 (using a sequence with the cloud - see figure 10) and in table 3 (using a sequence where no cloud has been added to the image - see figure 14). We also depict the true flows (figures 11 and 15), flows recovered by our method without validation (figures 12 and 16) and with validation (figures 13 and 17).

The first thing to note is that all methods generally perform better on the sequence without the cloud - it seems that the cloud has poor texture and so motion estimates in this region are generally unreliable. Considering the results for the cloudy sequence alone, we see that the results from our method (WLS) are clearly superior to the methods of Fleet and Jepson's and of Szeliski and Coughlan. Turning to the sequence without cloud, our method performs better than both that of Fleet and Jepson and of Black and Anandan.

One can also see, from these results, that the choice of the number of pairs of lines (m) used to approximated the LMedS, to remove outliers, does not have a great affect on the accuracy, providing the number m is greater than 10 or so.

Finally, we add a note of caution: a completely "fair" comparison is difficult to perform, as many authors clip part of the image region (usually the edges but also some of the sky

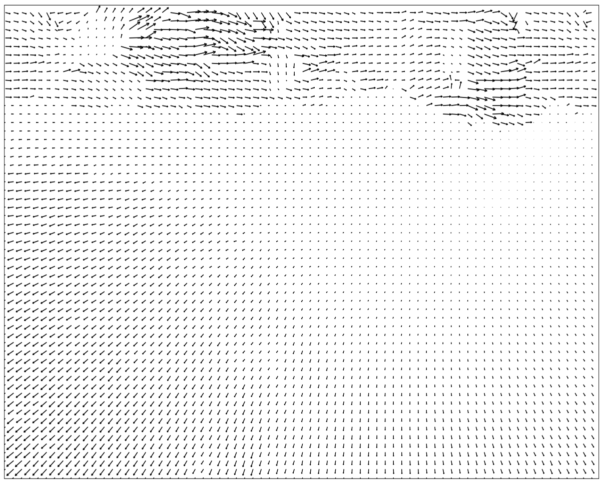

Figure 12: Yosemite Sequence WLS Flow (no validation) - with cloud

Visually, the flow field contains many errors in the cloud region.

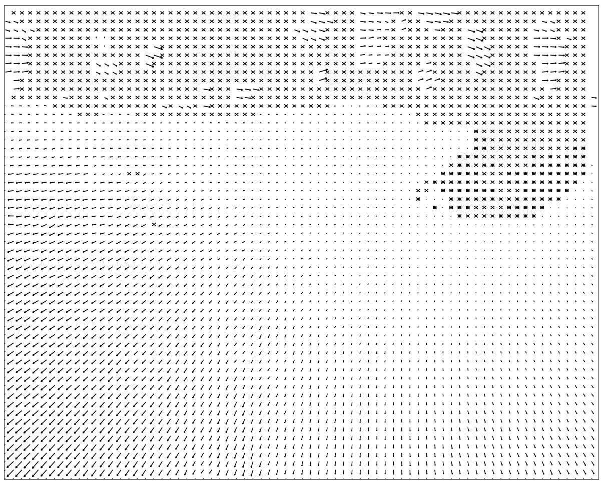

Figure 13: Yosemite Sequence WLS Flow $R^2 = 0.9$ - with cloud

The validation procedure has correctly identified many of the motion vectors in the cloud region as being unreliable. There are, however, still a number of undetected (relatively) poor estimates in this region (contributing to a slightly higher average error when compared with the same algorithm applied to the cloud free image sequence).

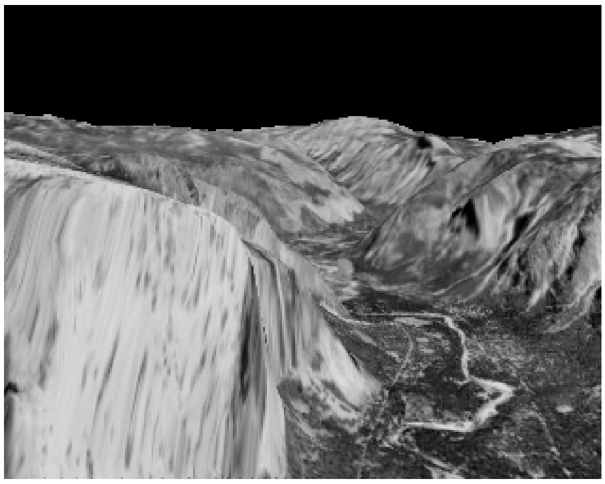

Figure 14: Yosemite Sequence - without cloud

Frame	Technique	Avg. Error	Std. Dev. Error	Density
Middle	Fleet and Jepson ($\sigma = 1.5$ $\tau = 1.25$)	5.28°	14.34°	30.6%
	Fleet and Jepson ($\sigma = 1.5$ $\tau = 2.5$)	4.63°	13.42°	34.1%
	Szeliski and Coughlan ($s = 2$, $T_e = 3000$)	2.19°	5.86°	23.1%
	Szeliski and Coughlan ($s = 2$, $T_e = 2000$)	3.06°	7.54°	39.6%
	WLS ($\sigma = 1.0$, 15x15, $m = 30$, without check)	7.39°	13.82°	100%
	WLS ($\sigma = 1.0$, 15x15, $m = 30$, $R^2 = 0.98$)	2.13°	3.22°	51.6%
	WLS ($\sigma = 1.5$, 15x15, $m = 30$, without check)	7.77°	16.2°	100%
	WLS ($\sigma = 1.5$, 15x15, $m = 30$, $R^2 = 0.98$)	2.06°	2.62°	64.0%
	WLS ($\sigma = 1.5$, 15x15, $m = 30$, $R^2 = 0.85$)	3.12°	6.54°	81.1%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, without check)	7.94°	16.1°	100%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, $R^2 = 0.98$)	2.23°	2.60°	66.3%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, $R^2 = 0.95$)	2.57°	4.03°	74.9%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, $R^2 = 0.90$)	2.99°	5.38°	78.4%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, $R^2 = 0.85$)	3.33°	6.35°	80.7%
	WLS ($\sigma = 2.0$, 25x25, $m = 30$, without check)	7.13°	14.2°	100%
	WLS ($\sigma = 2.0$, 25x25, $m = 30$, $R^2 = 0.98$)	2.11°	1.75°	58.2%

Table 2: Error analysis using Yosemite image sequence (with cloud)
The second column of entries determines the method applied to generate the row of error statistics. In our method (WLS), the numbers in brackets depict the size of the Gaussian smoothing (σ is the standard deviation of the filter), the size of local patch used (p), the number of pairs of lines used to approximate the LMedS (m), and the reliability threshold (R^2), in that order.

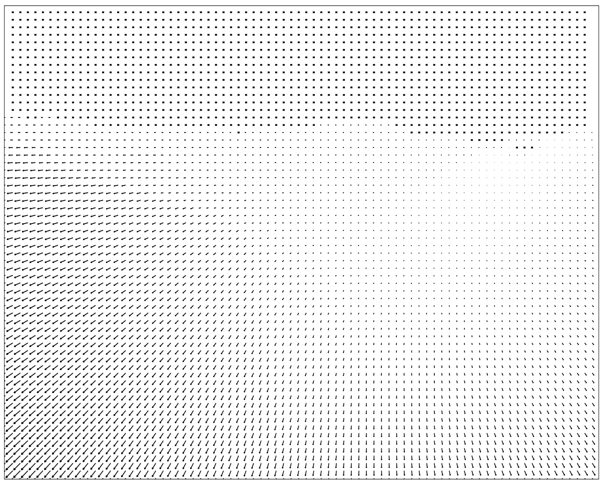

Figure 15: Yosemite Sequence Correct Flow - without cloud

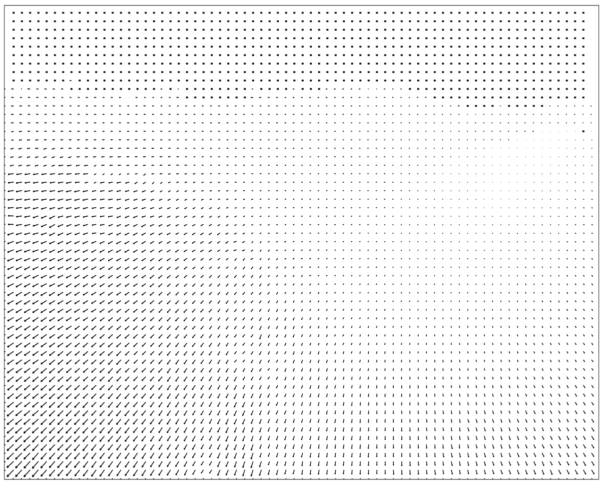

Figure 16: Yosemite Sequence WLS Flow (no validation) - without cloud
Visually, the flow field contains few detectable errors.

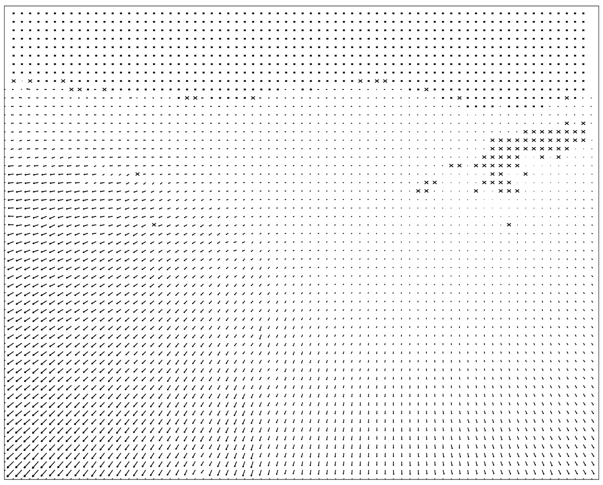

Figure 17: Yosemite Sequence WLS Flow $R^2 = 0.9$ - without cloud
The validation procedure has determined that a small patch (marked with $\times$) of velocities are unreliable.

Frame	Technique	Avg. Error	Std. Dev. Error	Density
9	Fleet and Jepson ($\sigma = 1.5$ $\tau = 1.25$)	4.0°	8.59°	30.0%
	Fleet and Jepson ($\sigma = 1.5$ $\tau = 2.5$)	3.42°	6.54°	33.52%
	Black and Anadan	4.46°	4.21°	100%
	WLS ($\sigma = 2.0$, 5x5, $m = 30$, without check)	3.80°	7.00°	100%
	WLS ($\sigma = 2.0$, 5x5, $m = 30$, $R^2 = 0.8$)	3.73°	6.92°	99.1%
	WLS ($\sigma = 2.0$, 5x5, $m = 10$, without check)	3.78°	6.99°	100%
	WLS ($\sigma = 2.0$, 5x5, $m = 10$, $R^2 = 0.8$)	3.71°	6.91°	98.3%
	WLS ($\sigma = 1.5$, 5x5, $m = 10$, without check)	3.75°	6.76°	100%
	WLS ($\sigma = 1.5$, 5x5, $m = 10$, $R^2 = 0.8$)	3.64°	6.61°	97.3%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, without check)	3.17°	6.46°	100%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, $R^2 = 0.8$)	3.18°	6.48°	99.1%
	WLS ($\sigma = 2.0$, 15x15, $m = 10$, without check)	3.17°	6.50°	100%
	WLS ($\sigma = 2.0$, 15x15, $m = 10$, $R^2 = 0.99$)	3.11°	7.12°	74.3%
	WLS ($\sigma = 2.0$, 15x15, $m = 10$, $R^2 = 0.9$)	3.18°	6.54°	98.1%
	WLS ($\sigma = 2.0$, 15x15, $m = 10$, $R^2 = 0.8$)	3.18°	6.48°	99.1%

Table 3: Error analysis using Yosemite image sequence (without cloud)
The second column of entries determines the method applied to generate the row of error statistics. In our method (WLS), the numbers in brackets depict the size of the Gaussian smoothing (σ is the standard deviation of the filter), the size of local patch used (p), the number of pairs of lines used to approximate the LMedS (m), and the reliability threshold (R^2), in that order.

region in the Yosemite sequence[1]) out before collecting error statistics. Often, they either do not mention this, or are too vague in their description for one to accurately repeat their results (or ensure comparable procedures are carried out for experiments on other methods).

5.3 Otte Image Sequence

This sequence is a real image sequence, recorded using a camera which translates toward a scene. The objects in that scene are stationary, except for a Marble block which is translating towards the left. A snapshot taken from this sequence is shown in figure 18 - for more details of the sequence see [28]. The scene contains many sharp discontinuities in both depth and motion.

The results of our experiments, are shown in table 4. We can clearly see, from these results that our method (WLS), particularly with large patch sizes, performs better than Fleet and Jepson's approach.

5.4 Translating and Diverging Tree

The diverging and translating tree sequences are two sequences created by a camera moving towards and parallel to (respectively) a poster picture of a tree. Thus, all of the image is essentially at the same depth (although there seems to be some slant) and the flow is rather simple (being essentially divergent in one and essentially parallel flow in the second). A snapshot indicative of the images in either sequence can be seen in figure 22.

The results produced by our experiments can be found in: table 5 (Diverging Tree Sequence) and table 6 (Translating Tree Sequence).

The results of experiments with the "Diverging Tree" sequence show that our method (WLS) produces results not markedly better than those of Fleet and Jepson's method. We believe that this is due to the fact that our flow model is simple (and we expect to be able to obtain the extra but small improvement necessary to match their result by incorporating an affine model into our approach). Certainly, since there are no discontinuities in motion,

[1]Indeed, if we clip 70 rows from the top (removing the sky region) and 7 pixels from the other boundaries (due to the Gaussian mask used to calculate the derivatives), similar to what we believe Black and Anandan did, we obtain an average error of 2.53° with standard deviation of 2.76° for all the remaining pixels in the cloudless Yosemite sequence (using $\sigma = 2$, 15×15, $m = 30$,withoutcheck).

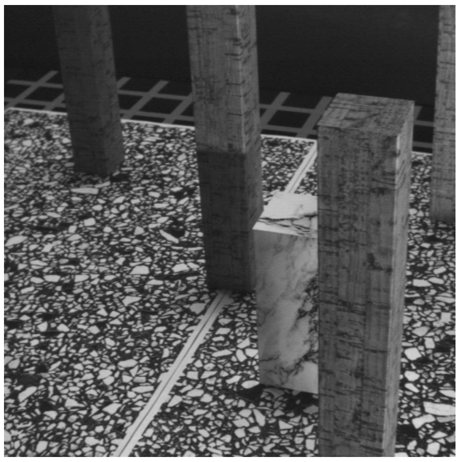

Figure 18: Otte Sequence
Snapshot taken from the Otte sequence.

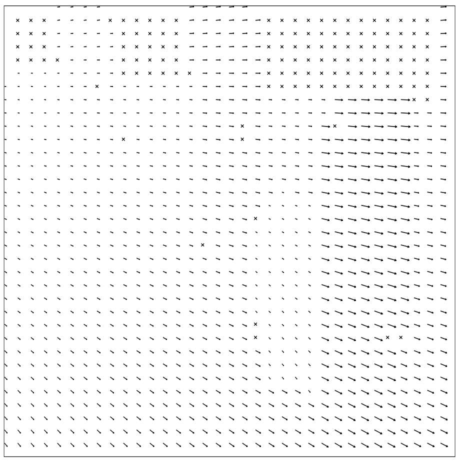

Figure 19: Otte Sequence - Correct Flow
Note that, unlike most test sequences with known velocity, there are actually patches of unknown velocity in this sequence (marked with $\times$ symbols).

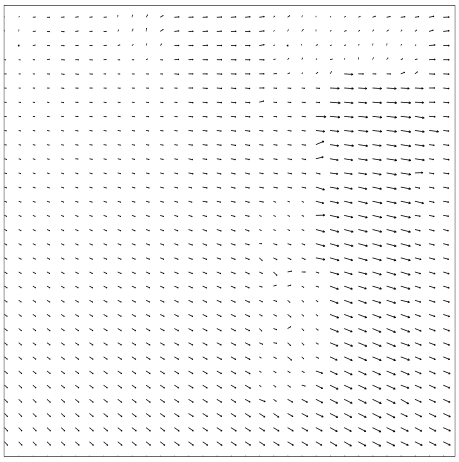

Figure 20: Otte Sequence - WLS Flow (no validation)

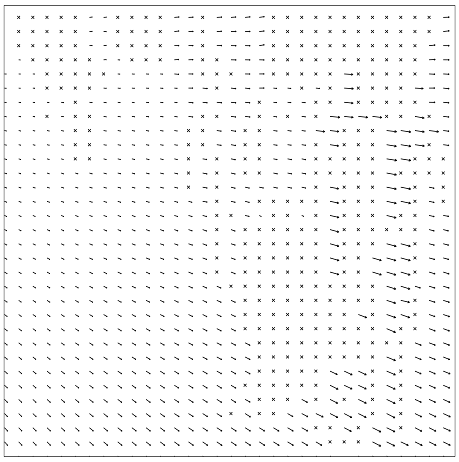

Figure 21: Otte Sequence - WLS $R^2 = 0.99$ Flow

Frame	Technique	Avg. Error	Std. Dev. Error	Density
35	Fleet and Jepson ($\sigma = 2.0$ $\tau = 1.25$)	2.08°	3.77°	50.6%
	Fleet and Jepson ($\sigma = 2.0$ $\tau = 2.5$)	2.56°	4.08°	57.1%
	Fleet and Jepson ($\sigma = 2.5$ $\tau = 1.25$)	2.05°	3.85°	55.8%
	Fleet and Jepson ($\sigma = 2.5$ $\tau = 2.5$)	2.53°	4.25°	62.2%
	WLS ($\sigma = 1.0$, 5x5, $m = 10$, without check)	7.40°	10.62°	100%
	WLS ($\sigma = 1.0$, 5x5,$m = 10$, $R^2 = 0.99$)	3.53°	3.87°	40.2%
	WLS ($\sigma = 1.0$, 15x15, $m = 10$, without check)	4.23°	5.69°	100%
	WLS ($\sigma = 1.0$, 15x15,$m = 10$, $R^2 = 0.99$)	2.70°	1.77°	47.7%
	WLS ($\sigma = 2.0$, 5x5, $m = 10$, without check)	6.81°	12.3°	100%
	WLS ($\sigma = 2.0$, 5x5,$m = 10$, $R^2 = 0.99$)	3.49°	7.54°	51.6%
	WLS ($\sigma = 2.0$, 15x15, $m = 10$, without check)	3.37°	6.5°	100%
	WLS ($\sigma = 2.0$, 15x15,$m = 10$, $R^2 = 0.99$)	1.47°	2.12°	58.2%
	WLS ($\sigma = 2.0$, 5x5, $m = 30$, without check)	6.83°	12.46°	100%
	WLS ($\sigma = 2.0$, 5x5,$m = 30$, $R^2 = 0.99$)	3.33°	7.11°	51.4%
	WLS ($\sigma = 2.0$, 15x15, $m = 30$, without check)	3.34°	6.48°	100%
	WLS ($\sigma = 2.0$, 15x15,$m = 30$, $R^2 = 0.99$)	1.5°	2.21°	59,2%
	WLS ($\sigma = 2.0$, 25x25, $m = 30$, without check)	2.58°	4.93°	100%
	WLS ($\sigma = 2.0$, 25x25,$m = 30$, $R^2 = 0.99$)	1.27°	1.82°	62.3%

Table 4: Error analysis using Otte image sequence

The second column of entries determines the method applied to generate the row of error statistics. In our method (WLS), the numbers in brackets depict the size of the Gaussian smoothing (σ is the standard deviation of the filter), the size of local patch used (p), the number of pairs of lines used to approximate the LMedS (m), and the reliability threshold (R^2), in that order.

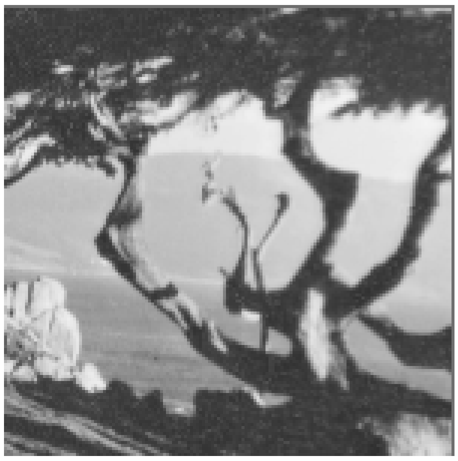

Figure 22: Tree Sequence
Snapshot taken from the Tree sequence. This is actually a photograph of a (flat) poster and the camera was moved in two different ways: approximately perpendicular to the surface of the poster (very approximately) to create a mainly divergent flow, and approximately parallel to the surface, to create the "translating flow".

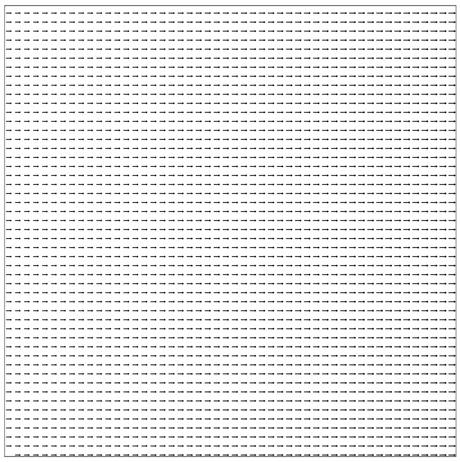

Figure 23: Tree Sequence (Translating) - Correct Flow
True Flow of the Translating Tree sequence.

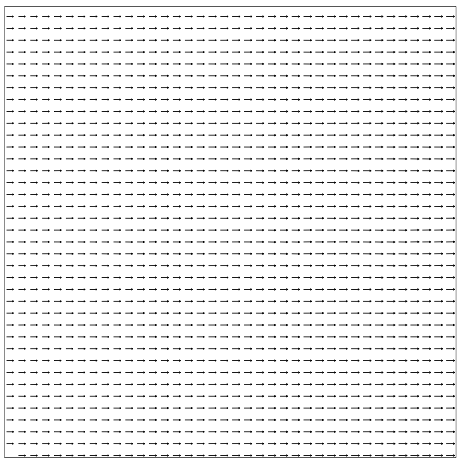

Figure 24: Tree Sequence (Translating) - WLS flow (no validation)
WLS Flow of the Translating Tree sequence.

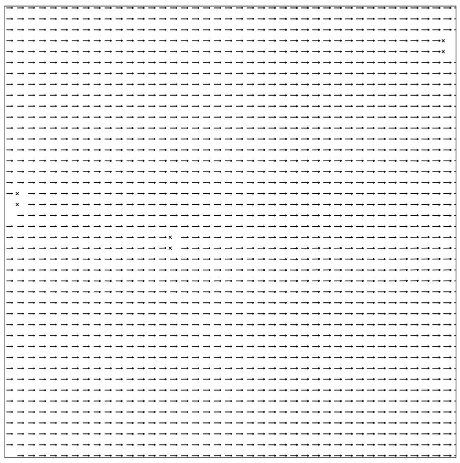

Figure 25: Tree Sequence (Translating) - WLS $R^2 = 0.99$ flow
WLS Flow ($R^2 = 0.99$) of the Translating Tree sequence.

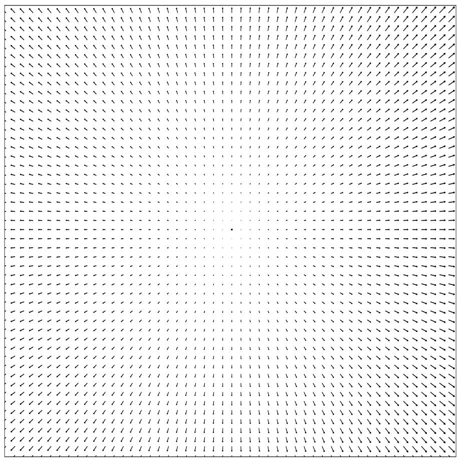

Figure 26: Tree Sequence
True flow of the Diverging Tree sequence.

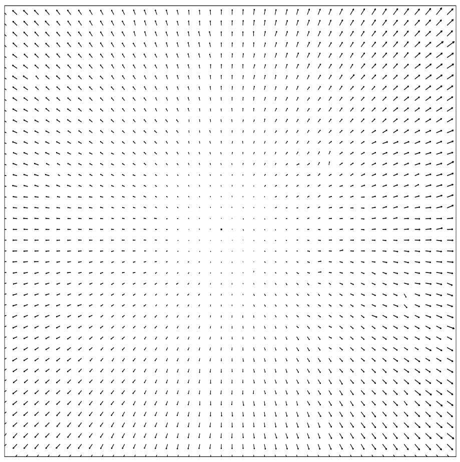

Figure 27: Tree Sequence (Diverging) - WLS flow
WLS Flow of the Diverging Tree sequence.

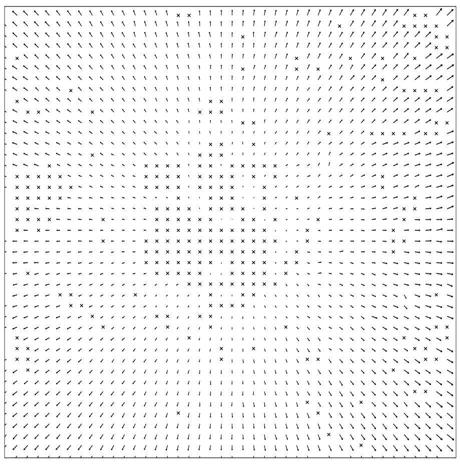

Figure 28: Tree Sequence (Diverging) - WLS $R^2 = 0.99$ flow
WLS Flow ($R^2 = 0.99$) of the Diverging Tree sequence.

Frame	Technique	Avg. Error	Std. Dev. Error	Density
20	Fleet and Jepson ($\tau = 1.25$)	0.80°	0.73°	46.5%
	Fleet and Jepson ($\tau = 2.5$)	0.99°	0.78°	61.0%
	WLS ($\sigma = 2.0$, 4x4, $m = 30$, without check)	2.95°	3.01°	100%
	WLS ($\sigma = 2.0$, 4x4,$m = 30$, $R^2 = 0.99$)	2.71°	2.69°	81.4%
	WLS ($\sigma = 2.0$, 5x5, $m = 30$, without check)	2.88°	2.78°	100%
	WLS ($\sigma = 2.0$, 5x5,$m = 30$, $R^2 = 0.99$)	2.66°	2.56°	82.8%
	WLS ($\sigma = 1.5$, 5x5, $m = 30$, without check)	2.48°	2.38°	100%
	WLS ($\sigma = 1.5$, 5x5,$m = 30$, $R^2 = 0.99$)	2.28°	2.2°	82.9%
	WLS ($\sigma = 2.0$, 6x6, $m = 30$, without check)	2.76°	2.42°	100%
	WLS ($\sigma = 2.0$, 6x6,$m = 30$, $R^2 = 0.99$)	2.57°	2.27°	84.8%
	WLS ($\sigma = 1.5$, 6x6, $m = 30$, without check)	2.35°	1.98°	100%
	WLS ($\sigma = 1.5$, 6x6,$m = 30$, $R^2 = 0.99$)	2.21°	1.87°	85.2%

Table 5: Error analysis using Diverging Tree image sequence
The second column of entries determines the method applied to generate the row of error statistics. In our method (WLS), the numbers in brackets depict the size of the Gaussian smoothing (σ is the standard deviation of the filter), the size of local patch used (p), the number of pairs of lines used to approximate the LMedS (m), and the reliability threshold (R^2), in that order.

Frame	Technique	Avg. Error	Std. Dev. Error	Density
20	Fleet and Jepson ($\tau = 1.25$)	0.23°	0.19°	49.7%
	Fleet and Jepson ($\tau = 2.5$)	0.32°	0.38°	74.5%
	Szeliski and Coughlan (local flow $n = 3$)	0.28°	0.23°	100%
	WLS ($\sigma = 1.0$, 15x15, $m = 10$, without check)	0.38°	0.34°	100%
	WLS ($\sigma = 1.0$, 15x15,$m = 10$, $R^2 = 0.999$)	0.33°	0.26°	96.3%
	WLS ($\sigma = 1.5$, 15x15, $m = 10$, without check)	0.32°	0.24°	100%
	WLS ($\sigma = 1.5$, 15x15,$m = 10$, $R^2 = 0.99$)	0.31°	0.24°	90.9%
	WLS ($\sigma = 1.5$, 15x15, $m = 30$, without check)	0.32°	0.23°	100%
	WLS ($\sigma = 1.5$, 15x15,$m = 30$, $R^2 = 0.99$)	0.31°	0.23°	99.5%
	WLS ($\sigma = 1.5$, 15x15,$m = 30$, $R^2 = 0.999$)	0.30°	0.23°	91.1%
	WLS ($\sigma = 1.5$, 25x25, $m = 10$, without check)	0.3°	0.19°	100%
	WLS ($\sigma = 1.5$, 25x25,$m = 10$, $R^2 = 0.99$)	0.3°	0.19°	100%

Table 6: Error analysis using Translating Tree image sequence
The second column of entries determines the method applied to generate the row of error statistics. In our method (WLS), the numbers in brackets depict the size of the Gaussian smoothing (σ is the standard deviation of the filter), the size of local patch used (p), the number of pairs of lines used to approximate the LMedS (m), and the reliability threshold (R^2), in that order.

the strengths of our approach, to deal with discontinuities, is unnecessary in this example. For the "Translating Tree" sequence, all methods report very low average errors. Again, the motion is extremely simple (no discontinuities and adequately modelled by locally (spatially) constant. This confirms our conjecture about the relative performance of our method on the "Diverging Tree" sequence. Our method must be employed with large patch sizes to give results comparable with those produced by Szeliski and Coughlan.

6 Conclusion

We have developed a robust method for solving a system of over-determined linear equations for the purpose of calculating optic flow.

The essence of the method is that we use an *approximate* Least median of Squares approach to identify outliers. This is particularly good at identifying and removing the effects of the breakdown of the motion consistency assumptions underlying all optic flow formulations (implicitly or explicitly, all formulations *must* use a form of this assumption to "beat the aperture problem"). The robustness, offered by the LMedS outlier detection, comes at only moderate computational cost: indeed, our method is faster than most other schemes claiming high accuracy. Having detected and removed outliers, we use a simple Least Squares approach, followed by a validation step (the latter, although probably useful in other situations, is certainly very necessary to detect situations where the local area does not contain one dominant motion).

We have implemented and tested the proposed method. It should be emphasised that the implementation contains many non-essential (to the central robust methods we have just outlined) aspects. For example, we have used a simple, patch-wise, assumption of uniform motion - this could be replaced with a more sophisticated affine motion model. Even so, despite being based upon such a very simple motion model, we are able to present results that generally out-perform or match every published method. Thus, our results, not only are of interest in that they demonstrate the effectiveness of our proposed robust methods; but they also seem to suggest that the simple motion models/assumptions are perhaps more reasonable than other works have suggested. Such considerations do not deny, however, that we expect even further improved performance by adapting our method to use more sophisticated motion models.

It should be emphasised that the complete scheme, as outlined here, can be elaborated upon in many ways. Many other innovations readily come to mind. One could develop a hierarchical implementation that would not only improve calculation times, but would also allow the method to cope better with larger motions (since the method uses a differential based approach, large motions can be problematic.) One could also develop a multi-pass approach: after identifying the outliers in the first pass, and solving for the motion associated with the majority population of pixels in a block, construct a second pass using only the "rejected" outliers, so as to recover other motions occurring within the block. We could also use M-estimators and Total Least Squares in the solution after outliers had been removed. These, and other innovations are the subject of ongoing work.

Finally, it should be stressed that, since the essence of our approach is that it is a robust method to solve an over-determined linear system of equations, the approach should be applicable to a wide variety of problems (including and beyond other problems drawn from computer vision research).

References

[1] D. J. Fleet and A. D. Jepson. Computation of component image velocity from local phase information. *International Journal of Computer Vision*, 5:77–104, 1990.

[2] C. Fennema and W. Thompson. Velocity determination in scenes containing several moving objects. *Computer Graphics and Image Processing*, 9:301–315, 1979.

[3] B. K. Horn and B. G. Schunck. Determining optic flow. *Artificial Intelligence*, 17:185–203, 1981.

[4] R. Szeliski and J. Coughlan. Hierarchical spline-based image registration. In *Proceedings CVPR'94, Seattle, June 1994*, New York, 1994. IEEE.

[5] M. Bober and J. Kittler. Estimation of complex multimodal motion: an approach based on robust statistics and hough transform. *Image and Vision Computing*, 12(10):661–668, December 1994.

[6] H-H Nagel. One the estimation of optical flow: relations between different approaches and some new ones. *Artificial Intelligence*, 33:299–324, 1987.

[7] T. Darrell and A. Pentland. Robust estimation of a multi-layered motion representation. In *Proceedings of IEEE workshop on visual motion, Princton, October 1995*, pages 172–178, New York, 1995. IEEE Press.

[8] M. J. Black and P. Anandan. The robust estimation of multiple motions: Paramteric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, January 1996.

[9] J. M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6(4):348–365, December 1995.

[10] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel. *Robust Statistics: The Approach Based On Influence Functions.* Wiley, New York, 1986.

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

[12] S-L Iu. Robust estimation of motion vector fields with discontinuity and occlusion using local outliers rejection. *Journal of Visual Communication and Image Representation*, 6(2):132–141, 1995.

[13] B. G. Schunk. Image flow segmentation and estimation by constraint line clustering. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(10):1010–1027, October 1989.

[14] P. Nesi, A. Del Bimbo, and D. Ben-Tzvi. A robust algorithm for optical flow estimation. *Computer Vision and Image Understanding*, 62(1):59–68, 1995.

[15] M. Bober and J. Kittler. Robust motion analysis. In *Proceedings of CVPR'94, Seattle, June 1994*, pages 947–952, New York, 1994. IEEE.

[16] M. Shizawa and K. Mase. Simultaneous multiple optical flow estimation. In *Proceedings of 10th International Conference on Pattern recognition*, pages 274–278, 1990.

[17] S. Wang, V. Markandey, and A. Reid. Total least squares fitting spatiotemporal derviatives to smooth optical flow field. In *Proceedings of the SPIE: Signal and Data Processing of Small Targets*, volume 1698, pages 42–55, 1992.

[18] J. Weber and J. Malik. Robust computation of optical flow in a multi-scale differential framework. *International Journal of Computer Vision*, 14:67–81, 1995.

[19] P. Meer, D. Mintz, A. Rosenfeld, and D. Y. Kim. Robust regression methods for computer vision: A review. *International Journal of Computer Vision*, 6(1):59–70, 1991.

[20] A. Mitiche. *Computational Analysis of Visual Motion*. Plenum, New York, 1994.

[21] S. VanHuffel and J. Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis*. SIAM, Philadelphia, 1991.

[22] J. Shi and C. Thomasi. Good features to track. In *Proceedings of CVPR'94*, pages 593–600, New York, 1994. IEEE Press.

[23] P. J. Rousseeuw. Least median of squares. *Journal of the American Statistical Association*, 79:871–880, 1984.

[24] J. M. Steele and W. L. Steiger. Algorithms and complexity for least median of squares regression. *Discrete Appl. Math.*, 14:93–100, 1986.

[25] H. Edelsbrunner and D. L. Souvaine. Computing least median of squares regression lines and guided topological sweep. *Journal of the American Satistical Association*, 85(409):115–119, 1990.

[26] H-H Nagel. Optical flow estimation and the interaction between measurement errors at adjacent pixel positions. *International Journal of Computer Vision*, 15:271–288, 1995.

[27] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Systems and experimental performance of optical flow techniques. *International Journal of Computer Vision*, 12:43–77, 1994.

[28] M. Otte and H-H Nagel. Optical flow estimation: Advances and comparisons. In *Proceedings of ECCV'94, Stockholm, Sweden, 2-6 May; 1994*, pages 51–60, 1994.

[29] T. O. Kvalseth. Cautionary note about R^2. *The American Statistician*, 39(4):279–285, 1985.

A Derivation of the Optic Flow Constraint

Consider a point in one image: as time evolves, this point traces out a curve in 3D (2D space plus time). We can parameterise that curve using time as the parameter. Thus we may write the curve as $s(t) = (x(t), y(t), t)$. In other words, given the time t (which image slice we take) we have the current position to the point as being $(x(t), y(t))$. We can suppress the formal dependence of x and y on t, in what follows. Now the image brightness (the intensity of the image point) is $I(x, y, t)$. We can take the total derivative of $I(x, y, t)$ and, assuming conservation of image brightness, we have:

$$0 = \frac{dI(x,y,t)}{dt} = \frac{\partial I(x,y,t)}{\partial x}\frac{dx}{dt} + \frac{\partial I(x,y,t)}{\partial y}\frac{dy}{dt} + \frac{\partial I(x,y,t)}{\partial t} \tag{9}$$

Since the components of the optic flow are $(u_x, u_y) = (\frac{dx}{dt}, \frac{dy}{dt})$ we have the OFC (equation 1).

B Outlier Threshold

In our method, having obtained an approximate solution, based on an approximate LMedS; we wish to assess the reliability of each constraint equation.

From equation 2, we have, for each patch, p equations or constraints. Rousseeuw and Leroy [11] give a good recipe for detecting outliers. We first calculate, for each constraint a residual r_i, then we calculate a scale factor s^0 according to:

$$s^0 = 1.4826(1 + \frac{5}{(p-2)})\sqrt{\underset{i}{med}\, r_i^2} \tag{10}$$

We then, for every constraint, associate a binary weight so that the weight is 0 for any constraint whose residual r_i is such that $|\frac{r_i}{s^0}|$ is greater than 2.5. Rather then using these weights, to directly reformulate the problem now as a (weighted) Least Squares problem, we go through one more step of scaling. This is because the original weights were chosen, according to equation 10, using the median involving the outliers. Since we now have a better idea of which are truly outliers, we calculate:

$$\sigma^* = \sqrt{\frac{\sum_{i=1}^{p} w_i r_i^2}{\sum_{i=1}^{p} w_i - p}} \tag{11}$$

and we, finally, reject those constraints for which the associated value $|\frac{r_i}{\sigma^*}|$ is greater than 2.5.

C Measure of Reliability

Although the LMedS technique has the highest possible breakdown point (50%) of all known robust estimators, it has the potentially fatal flaw in that is still produces an estimate, even if the number of outliers is more than 50%. Moreover, there are extreme cases where an image patch may not contain sufficient data (lack of texture) or data so badly corrupted (aliasing for example) for any estimate to be valid. Thus we still need to validate the estimate produced by our method.

A tool for the validation process can be modelled on "the coefficient of determination" [29]. The coefficient of determination, denoted R^2, has been defined for the Standard regression problem in at least nine different ways. The most well known of all the definitions is:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y})^2}{\sum_i (y_i - \overline{y})^2} \tag{12}$$

where $\hat{y}$ is the estimate of y provided by the regression fit and $\overline{y}$ is the mean of all of the data points y_i. The intuition behind this measure is that the mean, $\overline{y}$, is the best prediction (minimum variance of error, assuming nothing about the relationship between x_i and y_i in equation 7) [29].

For a robust form of Standard Regression, the following definition of R^2 has been used [29] [11]:

$$R^2 = 1 - \frac{\underset{i}{med}\, |y_i - \hat{y}|}{\underset{i}{med} \left\{ |y_i - \underset{i}{med}\, \overline{y}| \right\}} \tag{13}$$

However, although we are guided by analogy with the Standard Regression problem, we are interested in robust forms of Least Squares. In addition, the above R^2 measure requires the calculation of medians (a costly operation we would rather avoid). Moreover, the above measure is designed to be resistant to outliers: we want to use a measure that will tell us when the Least Squares fit is valid in terms of the data after we have rejected, *what we think are all of the outliers* - we do not want any statistic that is insensitive to the remaining outliers at this stage of validation.

These considerations lead us to define our own measure, which we also call R^2. We have found the following statistic, inspired by the form of the above measures, experimentally satisfactory:

$$R^2 = 1 - \frac{\sum_{i=1}^{p} w_i (d_i - \hat{d_i})^2}{\sum_{i=1}^{p} w_i (d_i - \overline{d_i})^2} \tag{14}$$

where $\overline{d_i}$ is the average value of the d_i and $\hat{d_i}$ is such that $a_{i1}\hat{u_x} + a_{i2}\hat{u_y} = \hat{d_i}$ for the estimates of flow $(\hat{u_x}, \hat{u_y})$ from our weighted least squares.