August 3, 1994 CS-94-06

An optical flow estimation algorithm that uses Gabor filters and affine model for flow.

Minas E. Spetsakis

Dept. of Computer Science
York University
4700 Keele Street
North York, ONTARIO
CANADA, M3J 1P3
minas@cs.yorku.ca, (416) 736-5053
FAX: (416) 736-5872

ABSTRACT

A general method for optical flow computation is presented that uses an affine model for the flow field. The method uses a hierarchy of filters and it is stable. It can work with or without stabilizers like smoothness terms. An extreme variety of filters can be used but we found the Gabor filters to be particularly good. Care has been taken to address the issue of the differentiation and preconditioning. The method was tested on real and synthetic images and compared with other methods. The performance was excellent for images with or without discontinuities, large or subpixel flow and rotating or dilating image sequences.

The support of the NSERC (App. No. OGP0046645) is gratefully acknowledged.

1. Introduction

The progress in the computation of optic flow during the past 15 years has been slow but steady. Up to now several algorithms have been proposed each one with strengths and weaknesses. There are several ways these all these algorithms can be classified. One way is to categorize them in gradient based (Lucas and Kanade [LUCA-84], Horn and Schunck [HORN-81], Nagel [NAGE-83], etc), region matching (Anandan [ANAN-89], Singh [SING-91], etc) and spatiotemporal filtering techniques (Heeger [HEEG-87], Fleet and Jepson [FLEE-90, FLEE-91, FLEE-92] and Jenkin and Jepson [JENK-90]). Another is to classify them in global variational methods (Horn and Schunck [HORN-81], Nagel [NAGE-83]) that are based on global minimization of some continuous functional over the image and local matching (Fleet and Jepson [FLEE-90], Lucas and Kanade [LUCA-84]) that work on a pixel or a small region to match intensity, phase etc. There are several other ways an algorithm can be characterized like hierarchical, regularization, phase based, etc, but the above are more relevant to our discussion.

The computation of optic flow presents several difficulties and this paper addresses some of the most important ones. One of the most discussed is the problem of the derivatives. It is well known that the numerical differentiation is a hard problem. In [KARA-91] where several differentiation schemes were analyzed, was shown that all methods (finite differences, polynomial fitting etc) fail for functions whose upper frequencies of the spectrum are very close to the Nyquist limit. All gradient based methods suffer from this because they depend on derivatives. In [BARR-93] a substantial improvement was reported when the finite differencing was substituted by more accurate derivatives. In later section we describe techniques that drastically improve the differentiation.

Another difficulty is related to the non uniformity of the flow. Most methods explicitly or silently assume that the flow can be considered uniform or smooth in every small region which is not true for realistic images. This affects any algorithm that matches regions without accounting for deformation [ANAN-89], and any algorithm that uses prefiltering (practically all). In a later section we analyze the phenomenon in detail but the basic intuition is this: The optic flow field is hardly ever uniform but in a small area it can be considered as a superposition of uniform motion and rotation, dilation and sheer, which is commonly referred to as affine deformation. The local Fourier spectrum of an area that undergoes rotation or dilation is transformed in a similar way. The dilation leads to shrinkage and the rotation to rotation. So if the image is filtered, then, as the different components of the spectrum of the image move their response will change as they hit different areas of the spectrum of the filter. This is shown in Fig. 1.1 where a cosine is shown to shrink. In the one frame the cosine is not filtered out, while in the next frame, where it shrunk and its frequency became higher, it was filtered out. This phenomenon appears whenever the flow is described by a first degree polynomial (affine flow) and we call it first order spectral instability. One solution to this is to isolate regions of the image where this phenomenon is dominant [FLEE-91], or isolate the regions of the spectrum that this phenomenon is generated [SPET-94a] or use an affine flow model like the one we describe in this paper. The latest approach has the advantage that it does not reject the information simply because it cannot use it, but it takes it into account.

Of course, one can have higher order spectral instabilities, when the variation of the flow is so high that it cannot be described meaningfully in a region of certain size without second order terms. In this case the local frequency of a cosine will change not only in time but in

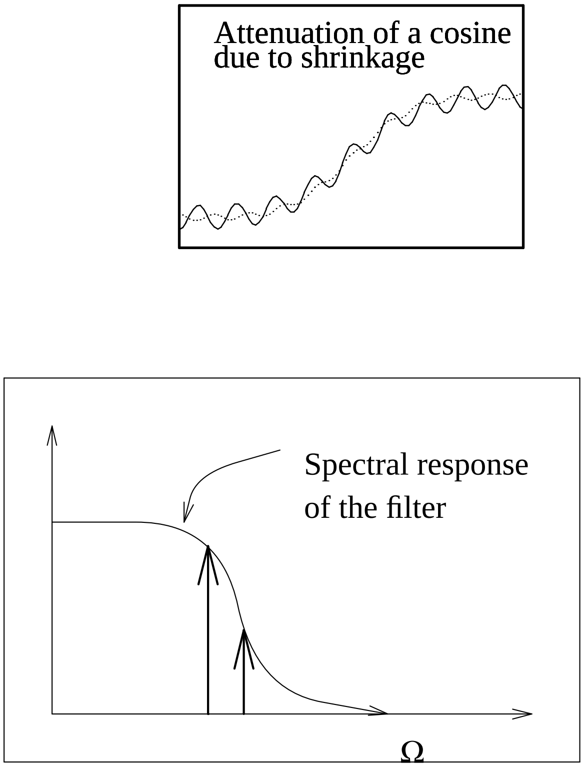


Figure 1.1 The figure shows the image of a cosine. The image shrunk as it moved, and the cosine went beyond the cutoff point of the filter and was attenuated.

space.

The third kind of difficulty, is related to the temporal alias. It means that the flow cannot be more than one pixel otherwise it is hard to take time derivatives. This is dealt with very successfully by methods that employ a hierarchical scheme [ANAN-89] or use several scales at the same time [JENK-90, XU-87, OLIE-93]. The method proposed here can both employ several scales in a single least squares minimization and use the result of the computation on a coarse version of the image as a guess to the next finer level.

The use of affine flow was to a large degree motivated by biological evidence [VERR-92, WERK-90] to model electrophysiological recordings and other observations. It has also been used to devise computational algorithms for flow estimation, e.g [VERR-87, OLIE-93] where the most common approach is to estimate both the flow and its derivatives as independent unknowns which are later used either directly or after interpolation. In this paper the formulation allows one to estimate the derivatives and the flow as interdependent parameters to get one consistent solution. A large variety of filters can be employed but Gabor filters seem to have advantages like more accurate computation of derivatives and better localization in the space and the frequency domain which results in better handling of discontinuities (the strip around the discontinuity that the algorithm is inaccurate is smaller).

The formulation that combines the estimation of the flow and its derivatives has several advantages. One is that the number of unknowns are still virtually two per pixel (ignoring the boundaries) because the derivatives are directly related to the flow. Another is that the derivatives are consistent with the flow and no extra step is needed to recover a single flow incorporating the information of the derivatives. A third is that the filters used do not have be circularly symmetric to reduce the order of the computation and last and most important is that the resulting equation is just a second order partial differential equation completely isomorphic (but with very different coefficients of course) to the ones derived by regularization methods like [NAGE-83].

The organization of the rest of the paper is as follows. Sec. 2 describes the classic approach using a locally constant model for the flow and the problems associated with it. Sec. 3 describes the affine approach and how it solves the problem of first order instabilities. Sec. 4 describes the derivation of the Euler equations and the problems with differentiation and how these problems are ameliorated with the use of complex Gabor filters. Other filters are also discussed. Sec. 5 describes experiments with real and synthetic images and final section summarizes the advantages of this approach and presents some concluding remarks.

2. Constant model of optic flow

Most flow estimation algorithms try to minimize a measure of the residual of the optic flow equation:

$$I_{err} = I_x u + I_y v + I_t \tag{2.1}$$

where I is the image intensity, the subscripts x, y and t denote the corresponding derivatives and u and v are the two components of the flow field we are trying to find. The most common measure of the residual is the Sum of Squared Errors (SSE).

But Eq. (2.1) has a few shortcomings that are well documented in the literature, like the fact that it is one equation but has two unknowns, the dependence on derivatives etc. Among the several ideas published so far, we choose the hierarchical approach [ANAN-89] and the integration of the equations in a small region [LUCA-84] that have been proven very good experimentally [BARR-93] and as we show immediately they have very nice generalizations.

Since we assume that the flow does not change much in a small region around x_0, y_0 we can write the optic flow equation as

$$I_{err}(x_0, y_0; x, y) = I_x(x, y)u(x_0, y_0) + I_y(x, y)v(x_0, y_0) + I_t(x, y) \tag{2.2}$$

and we seek a solution for u and v that makes the residual I_{err} as small as possible in a region around (x_0, y_0). If we minimize the sum of the squares of I_{err} in such a region then we get a method similar to Lucas and Kanade. But we try something more general. We minimize the sum of squares of linear combinations of I_{err} in Eq. (2.2) in this neighborhood.

$$SSE(x_0, y_0) = \sum_q \left\{ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_{err}(x_0, y_0; x, y) g_q(x - x_0, y - y_0) dxdy \right\}^2 =$$

$$\sum_q \left\{ \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_x(x, y) g_q(x - x_0, y - y_0) dxdy \right] u(x_0, y_0) + \right.$$

$$\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_y(x, y) g_q(x - x_0, y - y_0) dxdy \right] v(x_0, y_0) +$$

$$\left. \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_t(x, y) g_q(x - x_0, y - y_0) dxdy \right] \right\}^2$$

where g_q are the weights of the various linear combinations. So the sum of squares can be written

$$SSE(x_0, y_0) = \sum_q \left\{ I_x^{(g)} u + I_y^{(g)} v + I_t^{(g)} \right\}^2 \tag{2.3}$$

where the superscript (g) denotes convolution with the filter g_q.

The choice of filters g_q, determines the behavior of the algorithm. If the filters g_q are successively narrower lowpass filters for $q = 1..q_{\max}$ then we get a hierarchical algorithm. If the templates of the filters are shift templates, eg. they are zero everywhere except for one pixel that is one, so that convolution with them actually shifts the image, then we get Lucas and Kanade. Another choice is Gabor filters of various orientations and resolutions and if we use the phase of the Gabor filters instead, we get a variant of the phase based algorithms [FLEE-84, JENK-90].

The above analysis assumes that u and v do not vary much in a small region. This is a restrictive assumption in that requires the spectrum of the flow to be confined around the zero frequency. The obvious solution to avoid filtering does not help, because the filtering is introduced anyway as a noise reduction or image conditioning preprocessing step. To get a better solution we have to see what happens if we apply (2.3) on an image that is filtered with a filter g_q that has response $G_q(\omega)$. To do this, we follow one sinusoidal component of the image through time restricting ourselves to one dimension

$$I(x, t = 0) = \alpha e^{j\omega x}$$

where α is the corresponding Fourier coefficient. We assume that an affine model for motion is enough for a small region, so the image intensity becomes

$$I(x, t) = \alpha e^{j(\omega(t)x + \phi(t))}$$

where the phase shift ϕ is due to the uniform component of the motion and ω now depends on time to accommodate the shrinkage and dilation which is due to the non uniform component of the flow. The filtered version of the image is

$$I^{(g)}(x, t) = \alpha G_q(\omega(t)) e^{j(\omega(t)x + \phi(t))}$$

It is easy to verify that the optic flow equation on the unfiltered version of the image gives us

$$I_x dx + I_t dt = 0 \tag{2.4}$$

where $I_x = j\omega(t)I$ and $I_t = j(\omega'(t)x + \phi'(t))I$ and primes denote derivatives of functions of one argument. The flow of the unfiltered version of the image is essentially what we expect. But if we replace the intensity with its filtered version we don't get the same results. The derivatives of the filtered version of the image are

$$I_x^{(g)} = j\omega(t)\, G_q(\omega(t))\, I$$

$$I_t^{(g)} = G_q{}'(\omega(t))\, \omega'(t)\, I \;+\; jG(\omega(t)) \left(\omega'(t)x + j\phi'(t) \right) I$$

where primed symbols indicate derivative of one argument functions. If we plug them in the optic flow equation (2.4) we get

$$I_x^{(g)}u + I_t^{(g)} = G_q{}'(\omega(t))\omega'(t)I \tag{2.6}$$

which in general is not zero and it is the cause of first order instabilities. As noticed in [FLEE-91] the effects of the instabilities were more prominent when the amplitude of the derivatives of $I^{(g)}$ was crossing zero in which case the value of the r.h.s. of (2.6) dominates. In any case the size of the filter is limited by the size of the area that we can consider as having uniform motion.

3. Affine motion

But we can relax the restriction considerably, by using the assumption that the flow can be, more or less, described by an affine equation in a small neighborhood. This basic idea has been used in various forms in the past [BLAC-93, MEYE-92, CAMP-90]. The flow then takes the form

$$\begin{aligned} u(x,y) &= u(x_0,y_0) + u_x(x_0,y_0)\cdot(x-x_0) + u_y(x_0,y_0)\cdot(y-y_0) \\ v(x,y) &= v(x_0,y_0) + v_x(x_0,y_0)\cdot(x-x_0) + v_y(x_0,y_0)\cdot(y-y_0) \end{aligned}$$

This will allow the size of the filter (or the variation of the flow within a given area) to be considerably larger. Then

$$\begin{aligned} I_{err}(x_0,y_0;x,y) = {} & I_x(x,y)u(x_0,y_0) + I_x(x,y)u_x(x_0,y_0)\cdot(x-x_0) + I_x(x,y)u_y(x_0,y_0)\cdot(y-y_0) + \\ & I_y(x,y)v(x_0,y_0) + I_y(x,y)v_x(x_0,y_0)\cdot(x-x_0) + I_y(x,y)v_y(x_0,y_0)\cdot(y-y_0) + \\ & I_t(x,y) \end{aligned}$$

We now take a linear combination over a small region defined by function g_q to get the affine residual

$$
\begin{aligned}
&I_{aerr}(x_0, y_0) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_{err}(x_0, y_0; x, y) g_q(x - x_0, y - y_0)\, dx\, dy = \\
&I_x^{(g)}(x_0, y_0) u(x_0, y_0) + I_y^{(g)}(x_0, y_0) v(x_0, y_0) + \\
&I_x^{(gx)}(x_0, y_0) u_x(x_0, y_0) + I_y^{(gx)}(x_0, y_0) v_x(x_0, y_0) + \\
&I_x^{(gy)}(x_0, y_0) u_y(x_0, y_0) + I_y^{(gy)}(x_0, y_0) v_y(x_0, y_0) + \\
&I_t^{(g)}(x_0, y_0)
\end{aligned}
\qquad \textbf{(3.3)}
$$

where the superscripts mean convolution with the corresponding filter. The filter $g_q x$ comes from terms like

$$I_x^{(gx)}(x_0, y_0) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_x(x, y) \cdot (x - x_0) \cdot g_q(x - x_0, y - y_0)\, dx\, dy$$

Then the sum of squared absolute errors is

$$SSE = \sum_q \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (I_{aerr})^2 = \sum_q \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left| \left(I_x^{(g)} u + I_y^{(g)} v + I_x^{(gx)} u_x + I_y^{(gx)} v_x + I_x^{(gy)} u_y + I_y^{(gy)} v_y + I_t^{(g)} \right) \right|^2$$

We used absolute values because the filters can be complex valued, like Gabor filters. All we have to do now is minimize SSE for all u's and v's. The terms like $I_x^{(gx)}$ are convolutions of the derivative of the image with respect to x with the templates like $g_q x$. The Fourier transform of $g_q x$ is

$$\mathbf{F}\left[g_q x \right] = j \frac{\partial G_q}{\partial \omega_x} \qquad (3.4)$$

where j is the imaginary unit and G_q is the Fourier transform of g_q.

We introduce the first derivative of the flow to use the affine model in an analysis similar to the one in the last section.

$$u_x(x=0,t) = -\frac{\omega'(t)}{\omega(t)}$$

because $u(x,t) = \dfrac{j(\omega'(t)x + \phi'(t))I}{j\omega(t)I}$. Then we try to find a β such that

$$I_x^{(g)}u + \beta I_x^{(g)}u_x + I_t^{(g)} = 0$$

It is easy to see that

$$I_x^{(g)}u + \beta I_x^{(g)}u_x + I_t^{(g)} = -\beta j\omega(t)G(\omega(t))I\,\frac{\omega'(t)}{\omega(t)} + G'(\omega(t))\omega'(t)I$$

We can solve for β and we get

$$\beta = j\,\frac{G'(\omega(t))}{G(\omega(t))}$$

From Eq. (3.4) we can see that

$$\beta I_x^{(g)} = jG'I_x = I_x^{(gx)}$$

So the affine model of Eq. (3.3) can deal with the first order instabilities.

4. The minimization procedure

The objective now is to minimize the sum of squared errors of the affine optical flow equation.

$$SSE = \sum_q \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (I_{aerr})^2 = \sum_q \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left|\left(I_x^{(g)}u + I_y^{(g)}v + I_x^{(gx)}u_x + I_y^{(gx)}v_x + I_x^{(gy)}u_y + I_y^{(gy)}v_y + I_t^{(g)}\right)\right|^2$$

Using the standard techniques [HORN-86], we can get the Euler equations for this minimization (we show it here for only one filter for simplicity)

$$\begin{aligned}\rho_u &= I_x^{(g)} I^*{}_{aerr} - \frac{\partial}{\partial x}\left(I_x^{(gx)} I^*{}_{aerr}\right) - \frac{\partial}{\partial y}\left(I_x^{(gy)} I^*{}_{aerr}\right) = 0 \\ \rho_v &= I_y^{(g)} I^*{}_{aerr} - \frac{\partial}{\partial x}\left(I_y^{(gx)} I^*{}_{aerr}\right) - \frac{\partial}{\partial y}\left(I_y^{(gy)} I^*{}_{aerr}\right) = 0\end{aligned} \qquad \mathbf{(4.1)}$$

where the star superscript denotes complex conjugate. A common difficulty with such algorithms is the computation of derivatives. It is well known that the derivatives amplify the higher frequencies which are dominated by noise. But this is not the whole story. The derivatives are numerically hard even with synthetic images where noise is not a problem. In [KARA-91] is shown that the accuracy of the numerical differentiation depends on the method used. If a two tap filter is used (finite differences) the result is accurate for the lower frequency components of the signal (about one sixth of the spectrum) and less and less accurate for higher frequencies. The accuracy can be increased using more expensive filters (more taps). In effect using more taps one can increase the part of the spectrum that the computation is accurate.

But using more expensive filters will not solve all the problems. The main difficulty is introduced by terms like

$$\frac{\partial}{\partial x}\left(I_x^{(gx)} I^*{}_{aerr}\right)$$

where we take the derivative of what is essentially the product of three signals: $I_x^{(gx)}$ is a signal and $I^*{}_{aerr}$ is composed of sums of products of two signals: derivatives of the image and derivatives of the flow components. The width of the spectrum of the product of two real signals is more or less the sum of the widths of the factors of the product. To see this consider an image that is just a cosine $\cos \omega_1 x$. If we multiply it by another image that is also a cosine $\cos \omega_2 x$, then the result will contain the $\cos(\omega_1 + \omega_2)$. So the spectrum of the result will be wider than each of the components.

If, on the other hand we use Gabor filters the bandwidth could even decrease. A signal that has gone through convolution with a Gabor function

$$e^{j(k_x x + k_y y) + \frac{x^2+y^2}{2\sigma^2}}$$

can be written in the following form

$$s_1 = e^{j(k_x x + k_y y)} f_1(x, y)$$

where f_1 is a bandlimited signal. If we multiply two signals $s_1 s_2{}^*$ we get $f_1 f_2{}^*$ which with the proper choice of the Gabor parameters can have a narrower band than the original signals s if $\sigma|k| \geq 3$.

4.1. Iterative solution of the linear system

The numerical analysis literature contains a wide variety of techniques to solve large linear systems iteratively, each one suitable for particular kinds of problems. For our experiments we tried Jacobi iteration and Conjugate Gradient. We chose these two mainly because of the reasonable convergence rate (Jacobi needs more iterations and some lowpass filtering between iterations to avoid divergence), the simplicity of the algorithm and the fact that they can be cast in terms of high level image operations that easier to port to SIMD hardware (although Conjugate Gradient needs inner product computations that can be a problem on some SIMD architectures).

Whether we use Jacobi or Conjugate Gradient we need a preconditioner e.g. a crude approximation to the full linear system that is easy to solve. The easiest choice is the diagonal of the system, which is typically used with the Jacobi iteration. Since we have two components in the flow u and v that are closely related, it is better to use the 2×2 block diagonal.

The four elements of the 2×2 block (which are of course images) are

$$
\begin{aligned}
B_1[k,l] &= \frac{\partial \rho_u[k,l]}{\partial u[k,l]} = I_x^{(g)^2} + I_x^{(gy)^2}(|)d_1{}^2 + I_x^{(gx)^2}(-)d_1{}^2 \\
B_2[k,l] &= \frac{\partial \rho_u[k,l]}{\partial v[k,l]} = I_x^{(g)} I_y^{(g)} + I_x^{(gy)} I_y^{(gy)}(|)d_1{}^2 + I_x^{(gx)} I_y^{(gx)}(-)d_1{}^2 \\
B_3[k,l] &= B_2[k,l] \\
B_4[k,l] &= \frac{\partial \rho_v[k,l]}{\partial v[k,l]} = I_y^{(g)^2} + I_y^{(gy)^2}(|)d_1{}^2 + I_y^{(gx)^2}(-)d_1{}^2
\end{aligned}
$$

where d_1 is derivative template[†]. The template is squared element by element and then convolved with the squared images. The (|) operator is convolution with an 1-D template oriented vertically and (–) is horizontal convolution. But the result of this convolution may contain high frequencies due to the squaring of both the images and the templates. In a sense this runs contrary to the purpose of the introduction of the preconditioning, because instead of speeding up the convergence, it amplifies the higher part of the spectrum where the instabilities seem to come from, in a way independent of the underlying image. There is an easy remedy to this though. Just convolve it with a low pass filter. This way the high frequency problems are reduced and at the same time the preconditioner can play its role better trying to guess the solution to the whole problem. The determinant of every 2×2 block is a good confidence measure for the estimate we get for this pixel. So we find the flow of which pixels to trust by thresholding $B_1\, B_4 - B_2{}^2$.

[†] If we want a noise suppressing first derivative template we can integrate the noise suppressing part of the template in the function g and use the generic $1 \times N$ derivative template

4.2. Boundary conditions and discontinuities

The solution of partial differential equations depends on the boundary conditions: a set of equations that specify what happens at the border of the domain, in this case the flow field, where our equations become meaningless. In fact in some cases, like the Lagrange equation for electric fields, the solution depends almost exclusively on the boundary conditions. The same observation should not apply to flow field estimation because the value of the flow in the interior of the image should not depend on the choice of the boundary conditions although we cannot expect the flow to be accurate near the border.

The boundary conditions we used in this implementation are “periodic”. This means that the image is considered one of the tiles that cover the plane. Although other schemes are possible as well, the periodic boundary conditions are more convenient: just turn all the convolutions to periodic and during the iterations we do not have to worry about the border equations. The border is just a flow discontinuity.

One of the most difficult problems with optical flow if the one of discontinuities. The flow is not defined on a discontinuity and in practice it cannot be computed reliably on a narrow strip around it. This algorithm has the advantage that the width of the strip can be adjusted by changing the σ of the Gabor filter e.g. for a $\sigma = 2.0$ the strip is at most 10 pixels wide and in practice it is much narrower. The reason for this is explained in the experimental analysis subsection.

In this version of the algorithm the σ remains constant for every particular filter throughout the image (different filters have of course different σ). In the future, we plan to apply ideas in [SPET-94b] to vary the size of the Gabor based on the likelihood of a discontinuity.

4.3. Stabilization terms

As described so far, the algorithm does not contain any stabilization terms. The role of these terms is to keep numerical instabilities in check, so that at regions of no information, where the random noise dominates, the algorithm does not diverge. In a sense this is similar to the regularization terms used extensively in computer vision but we use a very small λ that has no effects in regions of sufficient information, thus it introduces no smoothness. In the experiments the effect of the stabilization terms was very small but since the cost of including them is negligible we can use it. All we have to do is add λ to the terms involving second derivatives of the flow. The rest is the same. Nagel type stabilization factors are slightly more complicated but the cost is similar.

4.4. Coarse to fine estimation

The algorithm can tolerate large displacements even when the image contains high frequency components. This is largely due to the ability of the Gabor filters to work on each region of the spectrum separately. But for very high displacements one has to use a coarse to fine strategy [ANAN-89]. An estimate of the flow is obtained by reducing the images and applying the algorithm on them. Then we use the guess to deform the second image using two dimensional quadratic or cubic interpolation (we did not find any appreciable difference between the two) and the residual flow was computed from them. This combined with the use of Gabor filters of multiple scales and orientations gave excellent results.

5. Experiments

The experiments were done using both synthetic and real images. There were four sets of images: the 6 dots, the rotating squares, the diverging tree, and the NASA sequence. The 6 dots move in various directions with anything between zero and 3 pixels. The image is not very smooth so problems with derivatives are accentuated. The rotating squares is a set of four squares containing textures with main frequencies varying from 4 pixels per cycle (half the Nyquist limit) to 8 pixels per cycle. The second image was produced by rotating the first image using cubic interpolation. The diverging tree sequence is a plane approaching the camera so the flow field is diverging (images 10 and 19). The NASA sequence has a few objects (a coke can, two pencils etc) on a table and the camera moves towards the center of the picture (images 1 and 5). The image flow is particularly small: between -1.5 and 1.5 pixels in each direction, which make it a good exercise in subpixel motion estimation.

We applied three different algorithms to each of them: Lucas and Kanade, and two versions of the affine algorithm described in this paper one that uses shift filters and one that uses Gabor filters. All the derivatives were calculated using 9 tap filters from [KARA-91]. The Lucas and Kanade was run with region size 11×11. The affine algorithm used Conjugate Gradient iteration. The version with shift filters (which can be thought of as the affine version of Lucas and Kanade) was run on a 11×11 region. The Gabor version used 8 complex filters arranged in two scales and four orientations, the scales differing by one octave, and the finest scale had a wave number $k = 1.5$ or about 4 pixels per cycle (the maximum value of the wave number is $k = \pi$).

The results were presented in three formats: One is the classic needle map that offers a general idea about the performance. The other is the deformable grid. The resulting optical flow is applied to a square grid where one can see easily the various glitches that usually result from some instability. The third is the inverse of the magnitude of the flow. This has the highest resolution, the glitches are really visible but the direction information is lost. Darker areas indicate large components of flow and bright areas small flow.

We compared our algorithm to Lucas and Kanade's because this algorithm turned out to be the second best in the comparisons in [BARR-93] with small difference from the best. Moreover it is simple and fast. We also experimented with our affine algorithm that uses simple shift filters to see which part of the improvement comes from the use of affine flow and which from the use of Gabor filters.

5.1. Experimental comparisons

The four images used in the experiments in this section are in Fig. 5.1.1

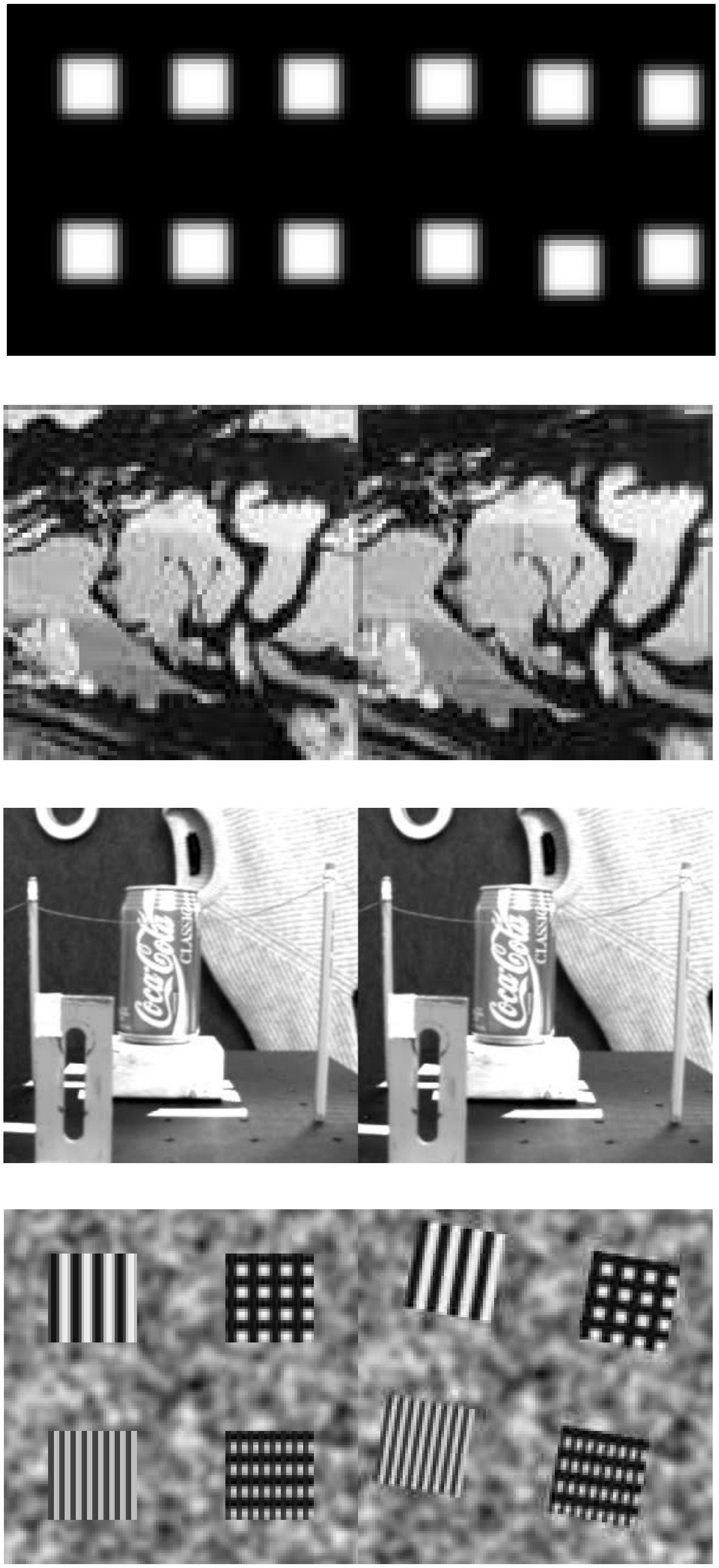


Figure 5.1.1 The four image pairs used for the experiments.

The results of applying the three algorithms to the “dots” sequence is in figures 5.1.2, 5.1.3, 5.1.4. The results for the “tree” and the “coke” sequence are in 5.1.5 to 5.1.10 and the “squares” are in 5.1.11 to 5.1.13. It is obvious that the affine algorithm using Gabors performs considerably better in all cases (no problems with derivatives, more stable output etc.)

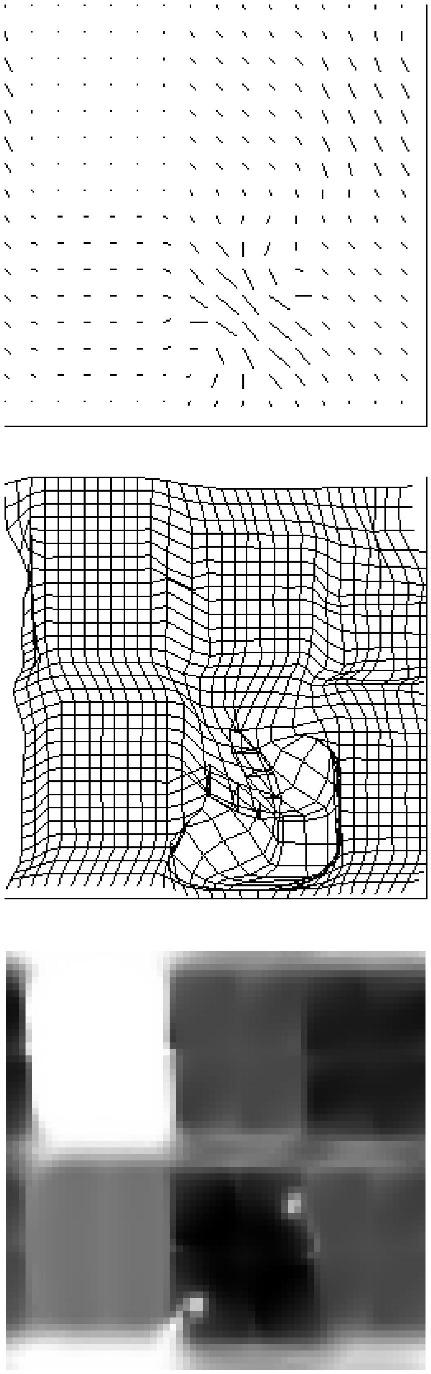


Figure 5.1.2 The result of the dots sequence with the Lucas and Kanade algorithm. The problems with the derivatives are obvious.

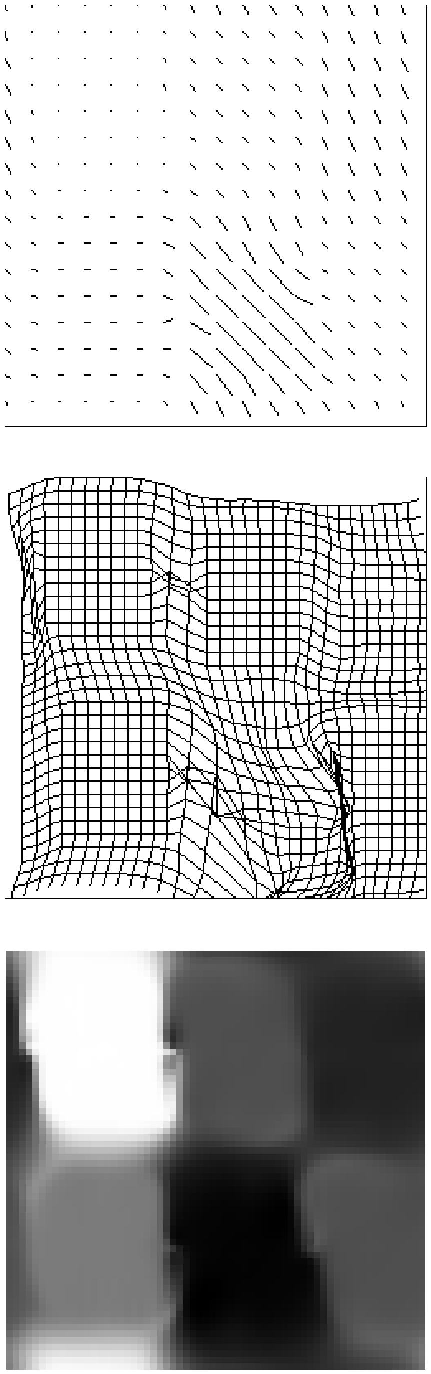


Figure 5.1.3 The result of the dots sequence with the shift filter affine algorithm. Again the derivatives create difficulties.

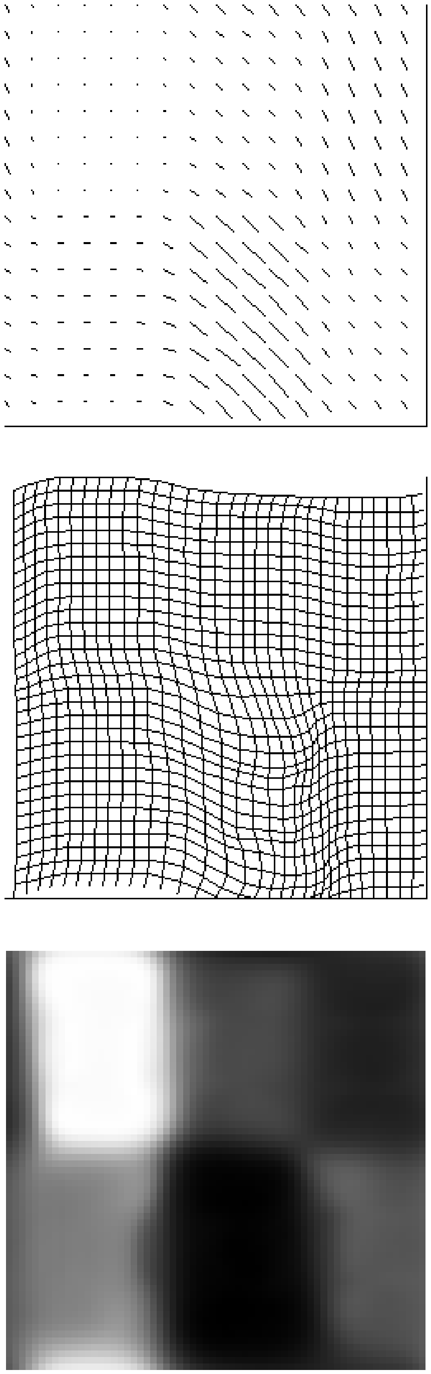


Figure 5.1.4 The result of the dots sequence with the Gabor filter affine algorithm. The behaviour is much better, there are no artifacts due to the derivatives.

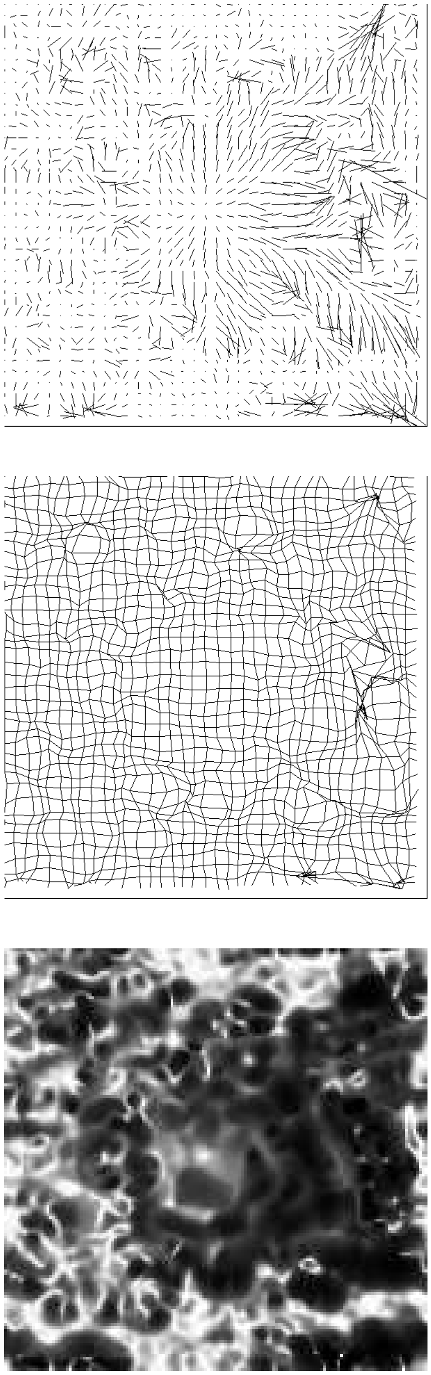


Figure 5.1.5 The result of the tree sequence with the Lucas and Kanade algorithm. The solution exhibits instabilities due to the large displacement. The non uniformity of motion makes the algorithm give useless results in large part of the image. What is worse is that the confidence measure is quite high in the parts of the image that the temporal aliasing creates bad artifacts.

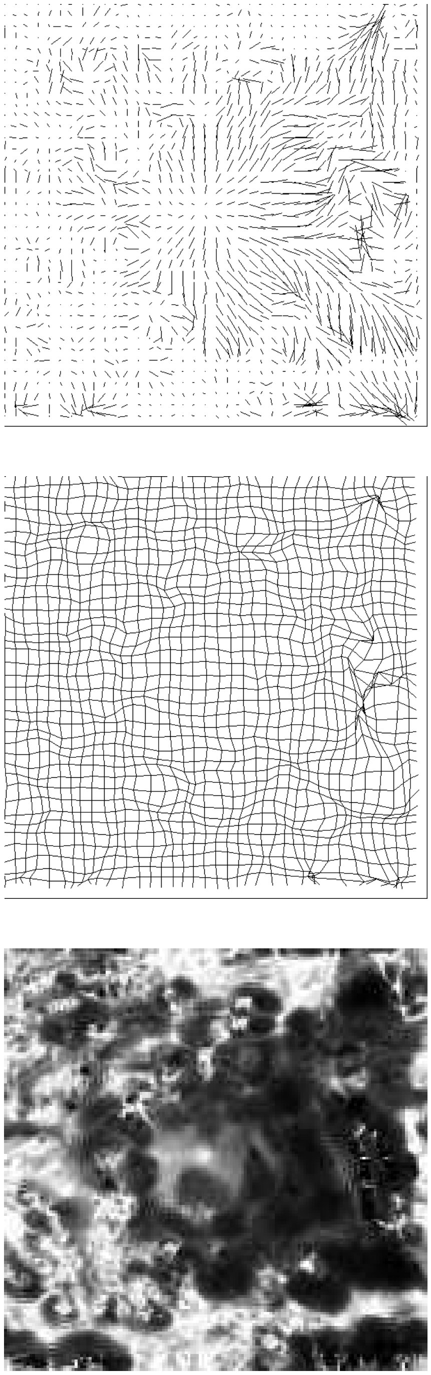


Figure 5.1.6 The result of the tree sequence with the shift filter affine algorithm. Fewer artifacts, but still unstable due to the difficulty with the derivatives.

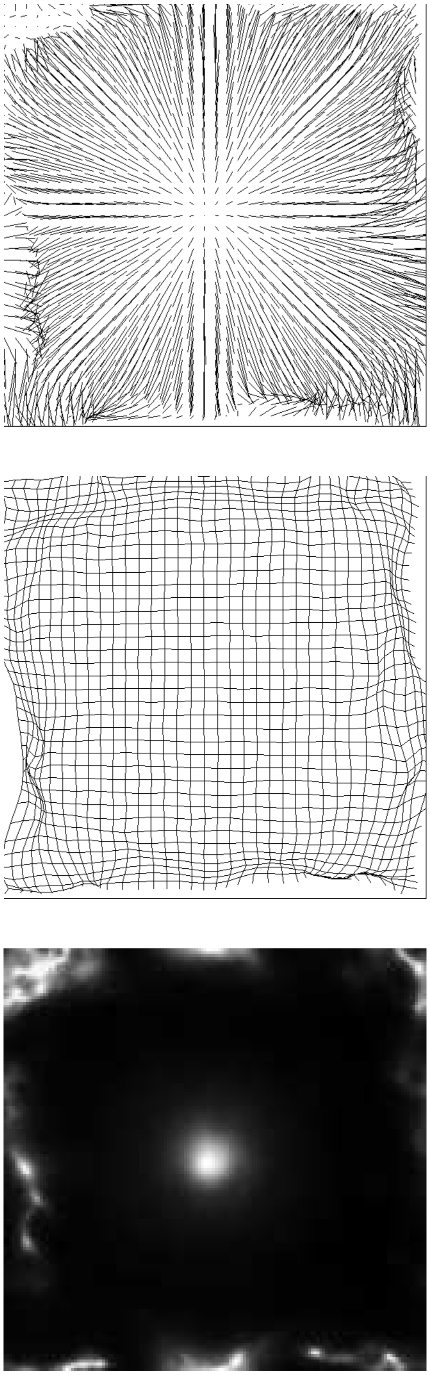


Figure 5.1.7 The result of the tree sequence with the Gabor filter affine algorithm. The only artifacts are near the borders (we used periodic boundary conditions). The rest is very close to the ideal.

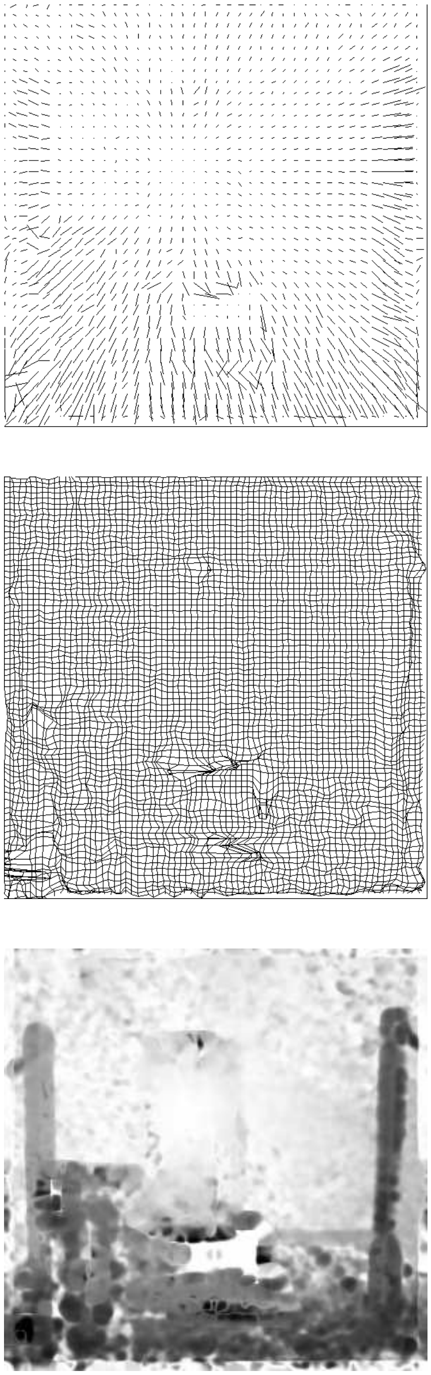


Figure 5.1.8 The result of the NASA sequence with the Lucas and Kanade algorithm. The results are acceptable, but the flow has large variations in places that it should be smooth.

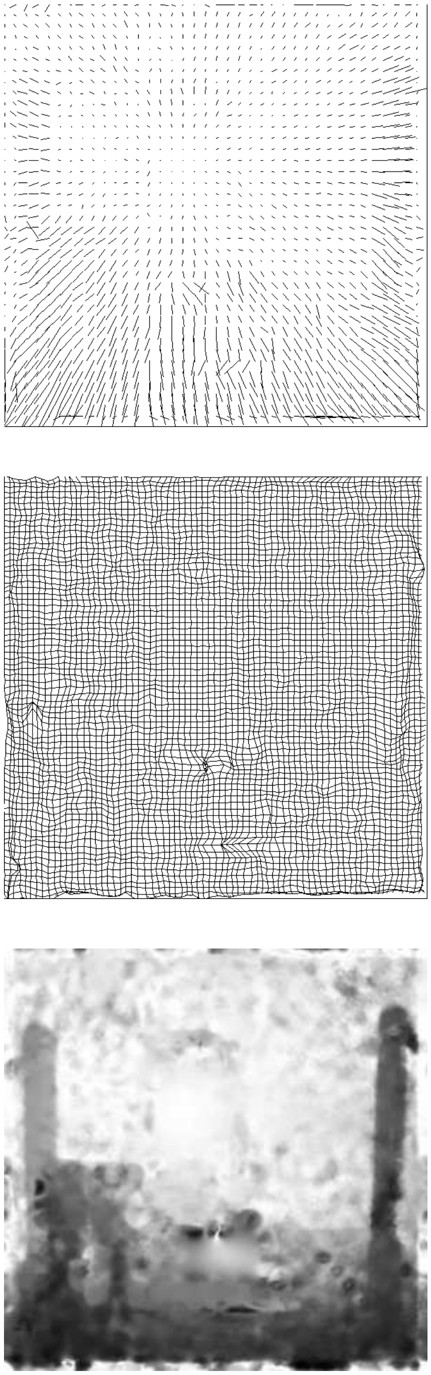


Figure 5.1.9 The result of the NASA sequence with the shift filter affine algorithm. The results are very good. The derivatives were no problem in this image so the behaviour is good.

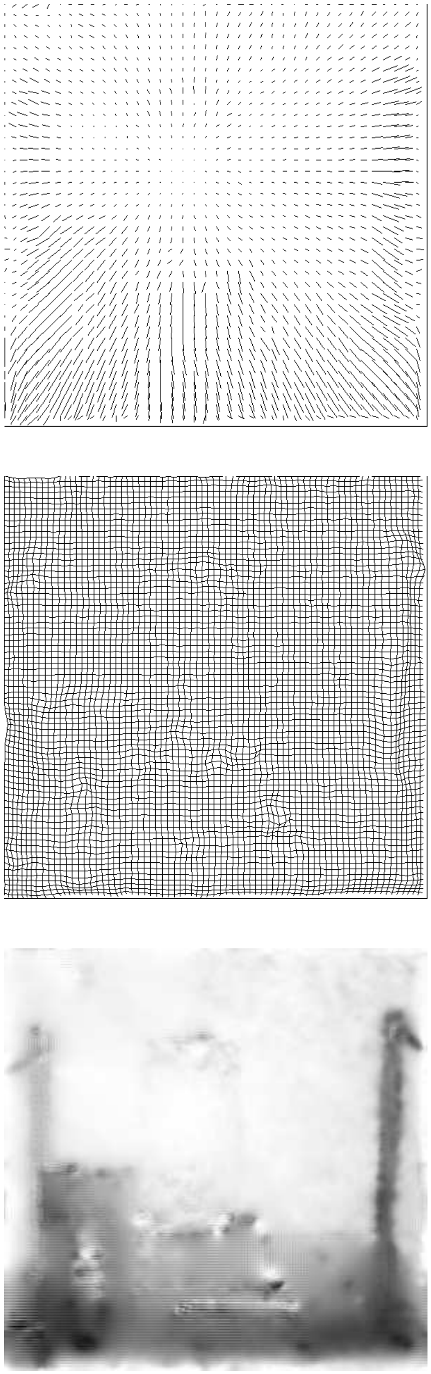


Figure 5.1.10 The result of the NASA sequence with the Gabor filter affine algorithm. Again the results are very good. The flow is much smoother in the areas where it should be smooth like the background and the sweater.

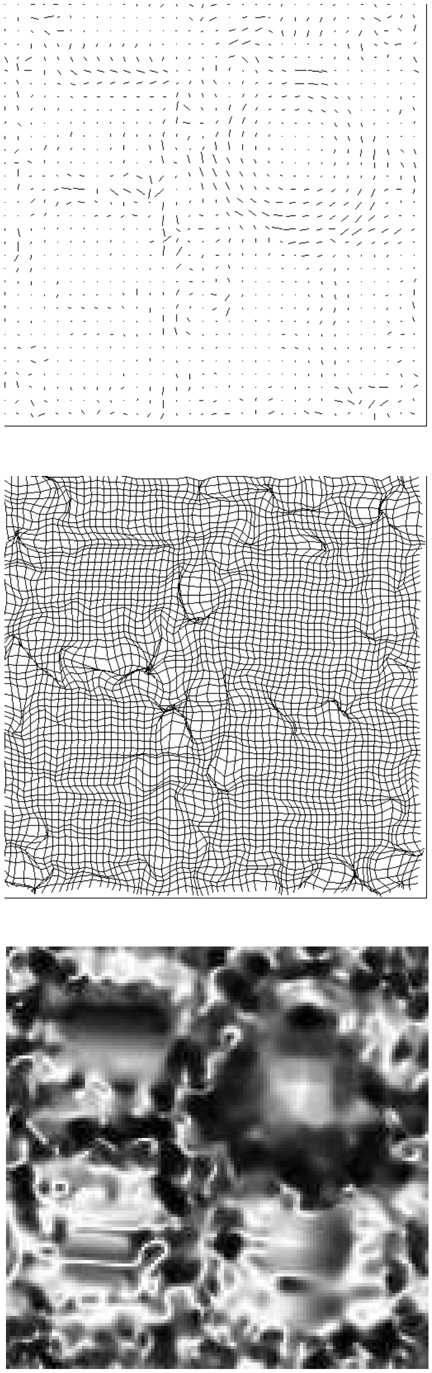


Figure 5.1.11 The result of the squares sequence with the Lucas and Kanade algorithm. The results are not acceptable even in near the center of rotation.

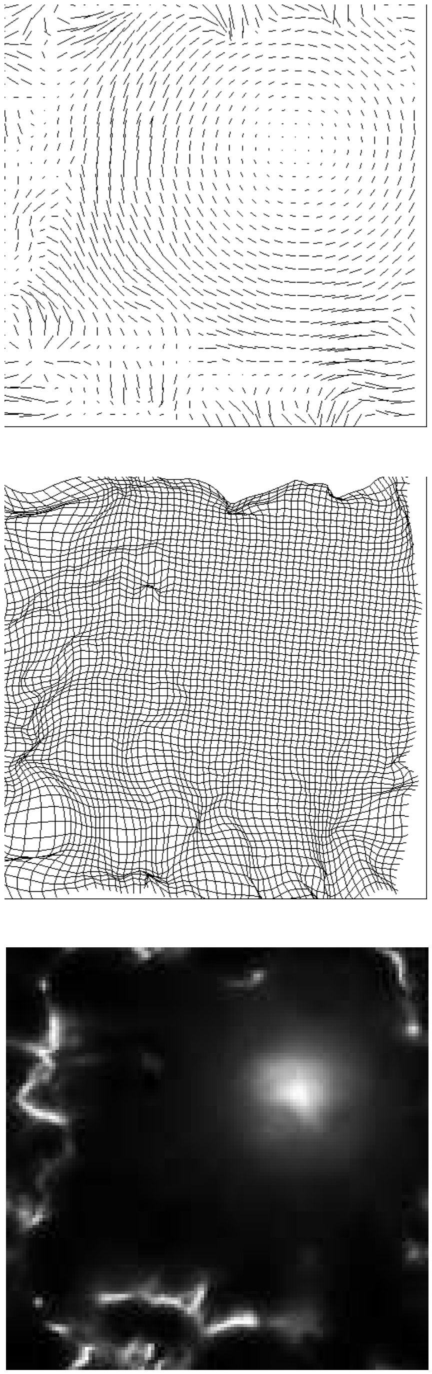


Figure 5.1.12 The result of the squares sequence with the shift filter affine algorithm. Some problems with the derivatives and temporal aliasing become obvious away from the center of rotation.

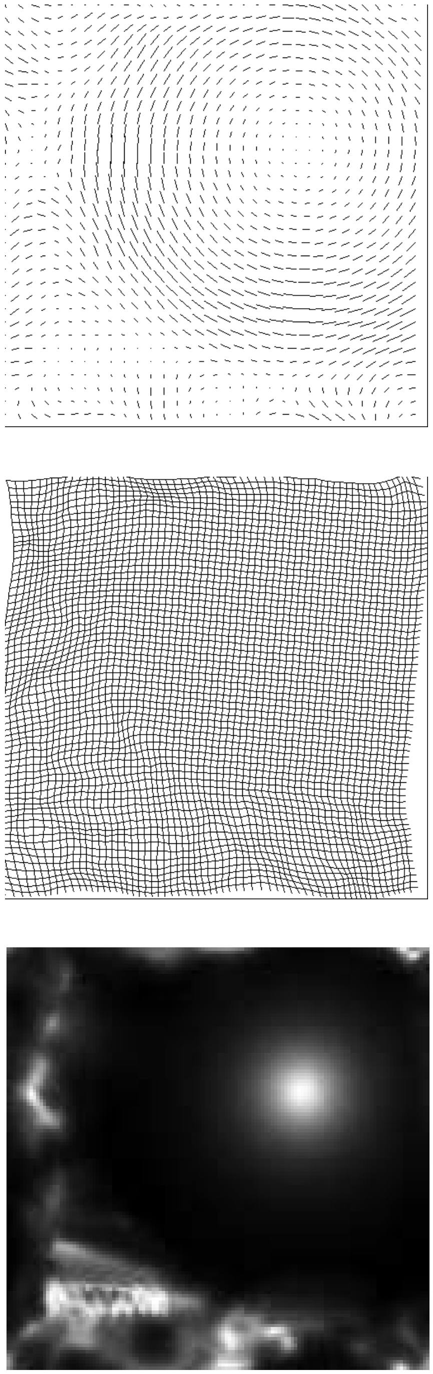


Figure 5.1.13 The result of the squares sequence with the Gabor filter affine algorithm. Again the results are very good even far away from the center of rotation.

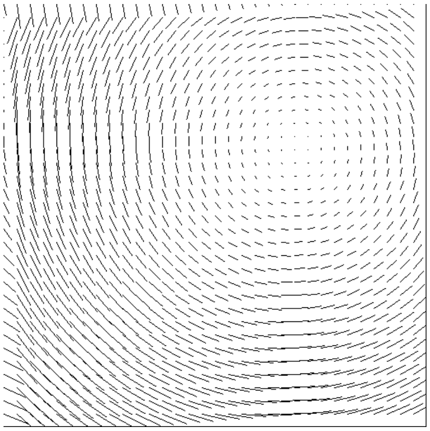
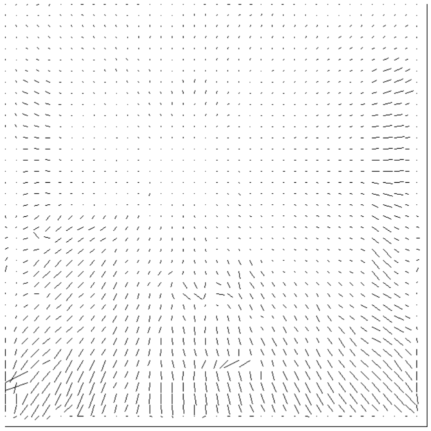


Figure 5.1.14 The ground truth for the squares.

5.2. Experimental analysis

While the performance of an algorithm depends mainly on the ideas that underlie the design, the implementation details and the refinement of the various “constants” that are unavoidable in every approach (the λ of Horn and Schunck, the size of the window of the region matching algorithms etc) can be the determining factors in an experimental comparison. The choice for instance between central time derivatives and derivatives at $t=0$ can make the difference between a random collection of vectors and a reasonable flow field. In this subsection we offer a set of random observations on the performance of the algorithm under various conditions.

The first is the inclusion of stabilizing terms. These terms have only a minor impact in the performance but they can make the algorithm more resilient in case that the choice of set of Gabor filters is bad. Figure 5.2.1 shows this for the NASA image with filters that were a bit too widely spaced appart in the frequency domain.



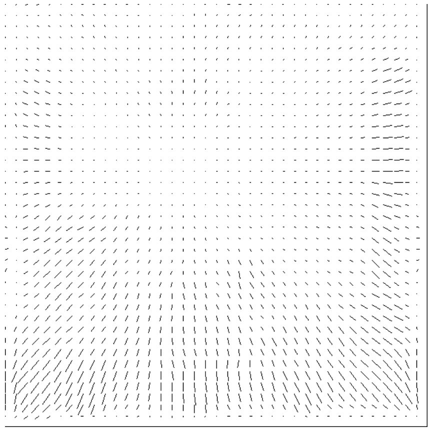


Figure 5.2.1 The stabilizers in the second needdlemap remove some instability in flow in the center of the image. The first image has no stabilizer and fewer filters that the optimum and the second has the same filters and rather strong stabilizers. All the instabilities occur at the borders of regions that have intensity 255.

Another issue is the use of various scales. There are two ways that a hierarchical strategy can be used in this algorithm. The one is to use filters of various scales in a single least squares optimization. This approach is simple and elegant and it is easy to predict the behavior theoreticaly. It is quite stable but also a bit oversmoothing, exactly as one would expect. The other approach is to shrink the images and apply the algorithm recursively on the finer ones. This has the advantage of efficiency and sharpness, but information obtained in the coarse scales can be completely ignored in the finer ones. In general it seems best to use two or three scales in each least squares combined with guesses obtained by the recursive approach. This seems to have beneficial effects in both the stability and efficiency. Further more the number of iterations goes down considerably.

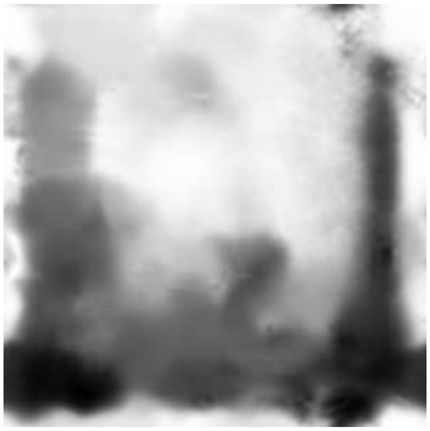


Figure 5.2.2 The image shows the inverse of the magnitude of the flow of the NASA sequence using the pure single least squares version of the algorithm. There is a decrease in sharpness compared with the corresponding picture from the previous section.

One unexpected benefit from the use of a guess to deform the second image and then compute the residual flow is that the sharpness on the discontinuities is increased. The reason is that the guess being mostly correct but localy oversmoothed on discontinuities, deforms the second image in such a way that the gap of the discontinuity is reduced. For images with large discontinuities, we found that calling the algorithm once more for the finest resolution only, results in dramatic increase of the sharpness.

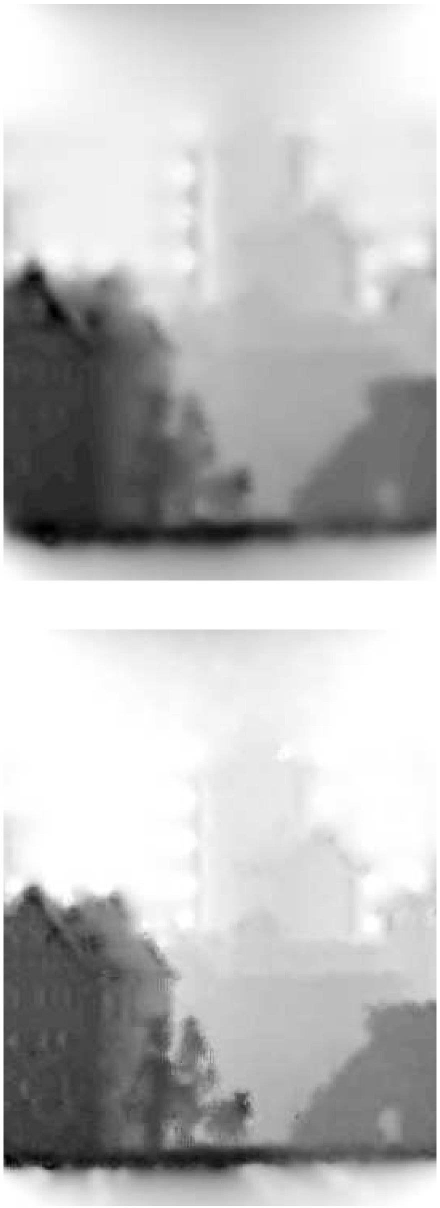


Figure 5.2.3 The top picture is the result of the recursive version of the algorithm on the castle seuqence (see below). The result was fed as a guess to the algorihtm (the finest level was used only). The pictures show the inverse of the magnitude of the flow, which is proportional to the depth map because the camera translates sideways. The flow varies between 20 and 30 pixels.

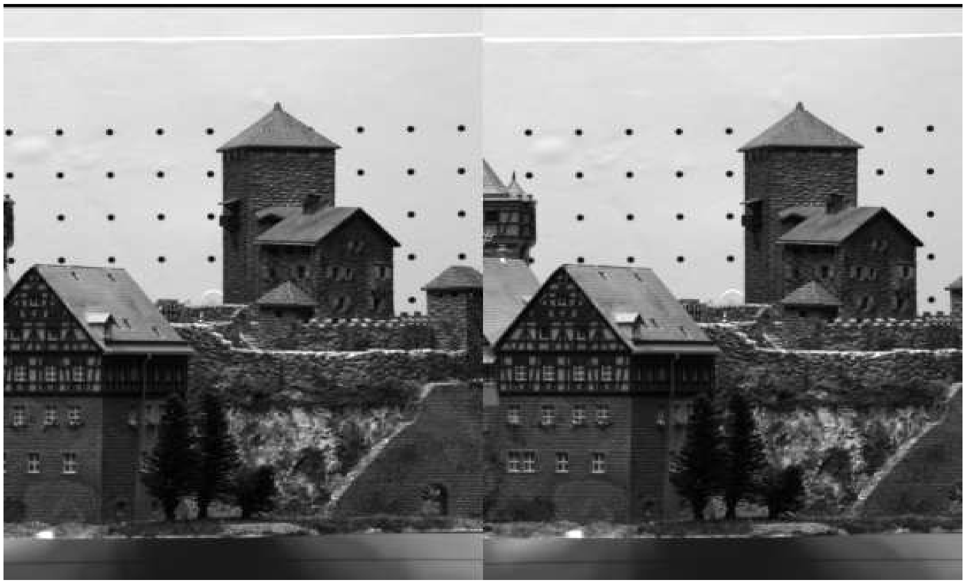


Figure 5.2.4 The two images of the castle sequence (numbers 3 and 4) cropped.

While the sequence in Figure 5.2.3 and Figure 5.2.3 has a huge flow that only a hierarchical algorithm can handle, in some of the images used, most notably the NASA coke sequence, the flow is for the most part subpixel. It turns out that the algorithm can handle subpixel flow equally well with large flow. We show this by magnifying the flow on a small window on the coke can of the NASA sequence.

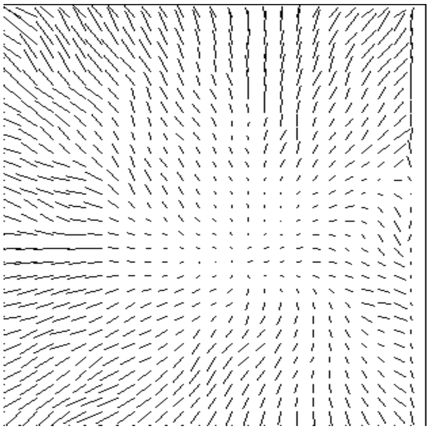


Figure 5.2.5 The flow in a small region on the coke can magnified 70 times. Since every cell is 2×2 pixels, a needle that is as big as the side of a cell is $\frac{1}{35}$ of a pixel. The largest flow vector in this image is about $\frac{1}{20}$ of a pixel.

6. Conclusions and future research

We presented an algorithm that is based on a locally affine model for flow. The algorithm can incorporate a wide variety of prefiltering as well as ideas from other algorithms. We proved that this algorithm is not affected by first order spectral instabilities.

There are several ideas along the lines of this formulation that are worth exploring. The first is to investigate discontinuities. One approach would be to use filters of varying support (area that the filter is non zero) and modify the weights according to the closeness to suspected discontinuities: After running the algorithm once, use the current estimate of the flow to adjust the weights of different filters, by increasing the weight of the wide filters when the second derivative of the flow is small and increasing the weight of the narrow filters when the second derivative is large.

Another approach to further improve the sharpness on the discontinuity is to classify every region of the image according to its probability to belong to a foreground or background object [JEPS-93] and use constraints from only one region.

7. Credits

The mathematical derivations for this paper were done with Maple. The implementation of this algorithm was done using MediaMath [SPET-94c]. Although the algorithm has fairly complicated structure, the implementation took only three days and the whole program was about three pages long.

The synthetic images were generated by MediaMath. The diverging tree sequence was offered by David Fleet and Leif Haglund. The coke can images are from the "NASA sequence" by Banavar Sridar and it was part of the standard image sequences of the Workshop on Motion 1991. The castle sequence is from the CIL at CMU.

Financial support was provided by NSERC.

References

ANAN-89.

P. Anandan, "A Computational Framework and an Algorithm for the Measurement of Visual Motion," *Intl' J. of Computer Vision* **2** pp. 283-310 (1989).

BARR-93.

J. L. Barron, D. J. Fleet, and S. S. Beauchemin, *Performance of Optical Flow Techniques,* RPL-TR-9107, Robotics and Perception Lab, Queen's University (July 1993).

BLAC-93.

M.J. Black and P. Anandan, “A framework for the robust estimation of optical flow,” *ICCV*, pp. 231-236 (1993).

CAMP-90.

M. Campani and A. Verri, “Computing Optical Flow from an Overconstraint System of Linear Equations,” *CVPR*, pp. 22-26 (1990).

FLEE-92.

D. J. Fleet, *The Measurement of Image Velocity,* Kluwer Academic Publishers, Norwell (1992).

FLEE-84.

D. J. Fleet and A. D. Jepson, “A Cascaded Filter Approach to the The Construction of Velocity Selective Mechanisms,” RBCV-TR-84-6, Department of Computer Science, University of Toronto (December 1984).

FLEE-90.

D. J. Fleet and A. D. Jepson, “Computation of component image velocity local phase information,” *Intl’ Journal of Computer Vision* **5** pp. 77-104 (1990).

FLEE-91.

D. J. Fleet and A. D. Jepson, “Stability of phase information,” *IEEE Workshop on Motion*, pp. 52-60 (1991).

HEEG-87.

D. J. Heeger, “A model for the extraction of image flow,” *ICCV*, pp. 181-190 (1987).

HORN-86.

B. K. P. Horn, *Robot Vision,* MIT Press (1986).

HORN-81.

B. K. P. Horn and B. G. Schunck, “Determining Optical Flow,” *Artificial Intelligence* **17** pp. 185-204 (1981).

JENK-90.

M. R. M. Jenkin and A. Jepson, “Response profiles of trajectory detectors,” *Trans. on Systems, Man, and Cybernetics* **19** pp. 1617-1622 (1990).

JEPS-93.

A. Jepson and M. Black, “Mixture models for optical flow computation,” *CVPR,* pp. 760-761 (1993).

KARA-91.

E. Karabassis and M. E. Spetsakis, *Families of templates for fundamental image operations,* York TR CS-91-07 (1991).

LUCA-84.

B. Lucas, *Generalized Image Matching by the Method of Differences,* PhD Dissertation, Dept. of Computer Science, Carnegie Mellon University (1984).

MEYE-92.

F. Meyer and P. Bouthemy, “Region based matching in an image sequence,” *ECCV,* pp. 476-484 (1992).

NAGE-83.

H. H. Nagel, “Displacement Vectors Derived from Second order Intensity Variations in Image Sequences,” *CVGIP* **21** pp. 85 - 117 (1983).

OLIE-93.

J. Oliensis and R. Manmatha, “Extracting affine deformations from image patches-I: Finding scale and rotation,” *CVPR*, pp. 754-755 (1993).

SING-91.

A. Singh, *Optic Flow Computation: A Unified Perspective,* IEEE Computer Society Press (1991).

SPET-94a.

M. E. Spetsakis, “Spectrum selective techniques for flow estimation,” *York University CS-ETR-94-01 (ftp wolf.yorku.ca)*, (1994a).

SPET-94b.

M. E. Spetsakis, “Optical flow estimation using discontinuity conforming filters,” *British Machine Vision Conference,* (1994b).

SPET-94c.

Minas Spetsakis, “MediaMath: A reasearch environment for vision research,” *Vision Interface,* pp. 118-126 (1994c).

VERR-87.

A. Verri and T. Poggio, “Against quantitative optical flow,” *ICCV*, pp. 171-180 (1987).

VERR-92.

A. Verri, M. Straforini, and V. Torre, “Computational aspects of motion perception in natural and artificial vision systems,” *Phil. Trans. R. Soc. Lond.* **B 337** pp. 429-443 (1992).

WERK-90.

P. Werkhoven and J. J. Koenderink, “Extraction of Motion Parallax Structure in the Visual System,” *Biological Cybernetics* **63** pp. 193-199 (1990).

XU-87.

G. Xu, S. Tsuji, and M. Asada, “A motion stereo system based on coarse-to-fine control strategy,” *PAMI* **9** pp. 332-336 (1987).