February 15, 1994 CS-ETR-94-02

An affine model for optical flow estimation

Minas E. Spetsakis

Dept. of Computer Science
York University
4700 Keele Street
North York, ONTARIO
CANADA, M3J 1P3

ABSTRACT

A model for optical flow is presented that assumes localy affine deformation of the moving image. It is shown that this formulation has clear advantages when the flow field is diverging, rotating or shearing. The problem is reduced to the solution of a partial differential equation of a simple form. Care has been taken to address the issue of the differentiation and preconditioning. The method was tested on real and synthetic images.

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

1. Introduction

The measurement of optical flow is one of the most fundamental and most studied problems in Computer Vision. It has also proven to be one of the hardest, and although it is now quite well defined, none of the algorithms proposed to date can claim that solved it. Despite that, there has been substantial progress. In a recent paper Barron, Fleet and Beauchemin gave a comprehensive quantitative account of this progress. Here we summarize the major developments of the field with emphasis on the aspects that we built on in this paper. The three major categories of optical flow algorithms are the gradient based (Lucas and Kanade [15], Horn and Schunck [11], Nagel [17], etc), the region matching (Anandan [2], Singh [19], etc) and the spatiotemporal filtering techniques (Heeger [18], Fleet and Jepson [8, 9, 6] and Jenkin and Jepson [12]. All these can also be classified along different lines like hierarchical, regularization, phase based, etc.

Every approach has advantages and disadvantages but there are a few problems that are shared by more than one approaches. One of the most discussed is the problem of the derivatives. It is well known that the numerical differentiation is unstable. In [14] several of these reasons are analyzed. The most important is the fact that all known methods (finite differences, polynomial fitting etc) fail for functions whose upper frequencies of the spectrum are very close to the Nyquist limit. And of course the presense of noise cannot help at all. All gradient based methods suffer from this because they depend on derivatives. In [3] 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 issue shared by many techniques is the fact that motion is not uniform. This affects any algorithm that matches regions without accounting for deformation [2], 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 [9], or isolate the regions of the spectrum that this phenomenon is generated [20] 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 space.

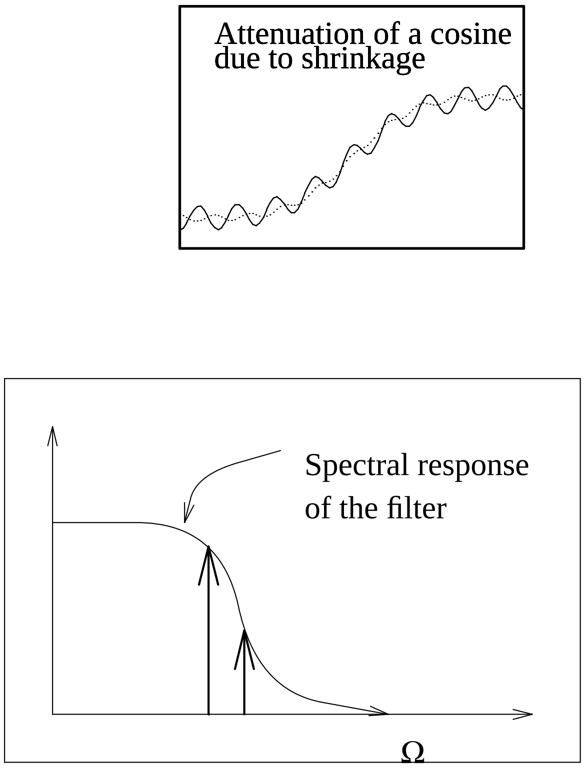
Attenuation of a cosine due to shrinkage
Spectral response of the filter
Ω

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.

It has been argued [22, 23, 1, 21] that only qualitative information could be extracted from optic flow, because the problem is inherently ill posed, and nevertheless the recent progress in active vision [5] indicates that we can do quite interesting things without accurate flow. It is definitely true that flow is a hard problem and the flow estimates cannot be considered anything more that a statistical estimate. But on the other hand the distinction between quantitative and qualitative information is only intuitive. More accurate qualitative information (like “object approaching from this direction”) can provide more accurate quantitative information (by, for example, fitting a divergent flow pattern in the area of the object that is approaching). The opposite is also true. For instance, the distance of two objects is corrupted by noise to a degree that makes them useless as estimates but if the estimates happen to be correlated, their difference might be considerably more accurate, to the degree that we can decide which is in front of the other, while we cannot estimate their absolute distance. Therefore a more accurate flow estimate can only help and be helped by qualitative estimation.

The organization of the rest of the paper is as follows. Sec. 2 describes the approach using a locally affine model for the flow. Sec. 3 describes the nature of the spectral instabilities and their effect on the optic flow equation and shows that using an affine model for flow these effects go away. Sec. 4 describes the derivation of the Euler equations and the problems with differentiation. It shows how to minimize the error introduced by the computation of derivatives and at the same time gives a framework to use ideas from other algorithms in conjunction with this one without any significant penalty in the efficiency. Sec. 5 describes experiments with real and synthetic images and final section summarizes the advantages of this approach and presents some concluding remarks.

2. Affine 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. The first, and the major, is that it is one equation per pixel for two unknowns per pixel. The second is that it involves derivatives, both in time and in space.

To solve this equation we need more information. Almost all algorithms use the assumption that the underlying flow has some structure, usually some form of smoothness or coherence. A smoothness assumption implies that the flow is smooth therefore we try to minimize a measure of roughness of the flow [17, 11]. A coherence assumption implies that all the points in a neighborhood move in a coherent way, which forms the basis of the region matching techniques [2, 19]. Other algorithms that use this assumption are Lucas and Kanade [15] that assumes that points within a region have almost the same displacement, and affine motion algorithms [13, 16, 4] that assume that motion can be described locally in terms of a first degree polynomial. Each of these approaches uses the same basic idea but each formalizes it in a different way with success in different aspects of the problem. The Horn and Schunck smoothness constraint is good because it can automatically interpolate in regions with no information. Lucas and Kanade seems to do very well experimentally [3] when the local deformations are not too high. Algorithms like Anandan's provide a neat way to integrate information from different levels of resolution using a coarse to fine strategy. And as we shall show in the next section the affine representation has a few more advantages that were not noticed before.

Here we opt for a coherence assumption, and as we show in the description of the algorithm later we can add a smoothness term at almost no cost.

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

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

If we minimize the sum of the squares of these equations in 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 Eq. (2.2) in this neighborhood.

$$\begin{aligned} SSE(x_0, y_0) = \sum_q \left\{ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_{err}(x_0, y_0; x, y) g_q(x - x_0, y - y_0) dxdy \right\}^2 = \\ \sum_q \left\{ \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_x(x, y) g_q(x - x_0, y - y_0) dxdy \right] u(x_0, y_0) + \right. \\ \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_y(x, y) g_q(x - x_0, y - y_0) dxdy \right] v(x_0, y_0) + \\ \left. \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_t(x, y) g_q(x - x_0, y - y_0) dxdy \right] \right\}^2 \end{aligned}$$

where g_q are the weights of the various linear combinations. 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 [7, 12].

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. We can relax it considerably by assuming that the flow can be, more or less described by an affine equation in a small neighborhood.

$$u(x,y) = u(x_0,y_0) + u_x(x_0,y_0)\cdot(x-x_0) + u_y(x_0,y_0)\cdot(y-y_0)$$
$$v(x,y) = v(x_0,y_0) + v_x(x_0,y_0)\cdot(x-x_0) + v_y(x_0,y_0)\cdot(y-y_0)$$

This will allow the spectrum of the flow 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

$$\begin{aligned} & I_{aerr}(x_0,y_0) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_{err}(x_0,y_0;x,y)g_q(x-x_0,y-y_0)\,dx\,dy = \\ & I_x^{(g)}(x_0,y_0)u(x_0,y_0) + I_y^{(g)}(x_0,y_0)v(x_0,y_0) + \\ & I_x^{(gx)}(x_0,y_0)u_x(x_0,y_0) + I_y^{(gx)}(x_0,y_0)v_x(x_0,y_0) + \\ & I_x^{(gy)}(x_0,y_0)u_y(x_0,y_0) + I_y^{(gy)}(x_0,y_0)v_y(x_0,y_0) + \\ & I_t^{(g)}(x_0,y_0) \end{aligned} \tag{2.3}$$

where

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

$$I_y^{(g)}(x_0, y_0) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_y(x, y) g_q(x - x_0, y - y_0)\, dx\, dy$$

$$I_t^{(g)}(x_0, y_0) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I_t(x, y) g_q(x - x_0, y - y_0)\, dx\, dy$$

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

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

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

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

Then the sum of square errors is

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

All we have to do now is minimize SSE for all u's and v's. Before we go to the next section, we'll have a look into the terms like $I_x^{(gx)}$. This is a convolution of the derivative of the image with respect to x with the template $g_q x$. The Fourier transform of $g_q x$ is

$$\mathbf{F}\left[g_q x\right] = j\,\frac{\partial G_q}{\partial \omega_x} \tag{2.4}$$

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

3. First order spectral instabilities

As we argued before the affine model cures the first order instabilities. We now show it in more detail. To simplify things we restrict ourselves to one dimension. The generalization to two dimensions is easy.

We analyze the image at $t = 0$ into its Fourier components

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

where E is the Fourier transform of the image. Every component of the image, under affine flow undergoes a transformation that is composed of a phase shift (due to the uniform component of the flow) and a frequency shift (due to the dilation). So

$$I(x, t) = \int E(\omega(t = 0)) e^{j(\omega(t)x + \phi(t,\omega))} d\omega \tag{3.1}$$

where ϕ is the phase shift and ω is now a function of time. In Eq. (3.1) we don't take the Fourier at $t \neq 0$ but apply the affine transformations on the components of the Fourier at $t = 0$. We now concentrate on one component of the spectrum only, so Eq. (3.1) becomes

$$I(x, t) = e^{j(\omega(t)x + \phi(t))} \tag{3.2}$$

The filtered version of the image is

$$I^{(g)}(x, t) = G_q(\omega(t)) e^{j(\omega(t)x + \phi(t))} \tag{3.3}$$

where G_q is the Fourier transform of the filter g_q. Then the optic flow equation on the unfiltered version of the image gives us

$$I_x dx + I_t dt = 0 \tag{3.4}$$

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. Then the flow u is

$$u(x=0,t=0) = -\frac{dx}{dt} = -\frac{I_t}{I_x} = -\frac{\phi'(0)}{\omega(0)} \tag{3.5}$$

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

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

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

If we plug them in the optic flow equation (3.4) we get

$$I_x^{(g)}u + I_t^{(g)} = G_q{}'(\omega(t))\omega'(t)I \tag{3.6}$$

which in general is not zero and it is the cause of first order instabilities. As noticed in [9] 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 (3.6) dominates. We introduce the first derivative of the flow to use the in affine model

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

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. (3-1.4) we can see that

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

So the affine model of Eq. (3-1.3) 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 [10], we can get the Euler equations for this minimization (we show it here for only one filter for simplicity)

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

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 [14] 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 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. The things only get worse if we multiply three signals. If we assume that our method of differentiation is accurate for half the spectrum and the three factors of the product have similar bandwidth w, then w cannot be more than one sixth of the whole spectrum. For 2-D signals, this means that we can only use the $\frac{1}{36}$ of the spectrum.

The solution, apart from increasing the resolution, is to use the rule for product differentiation and break it, as in

$$\frac{\partial a(x,y)b(x,y)}{\partial x} = \frac{\partial a(x,y)}{\partial x} b(x,y) + a(x,y) \frac{\partial b(x,y)}{\partial x} .$$

This way we take the derivative of each signal separately. So we first break the product of the two signals and Eq. (4.1) becomes

$$\rho_u = \left(I_x^{(g)} - I_{xx}^{(gx)} - I_{xy}^{(gy)} \right) I_{aerr} - I_x^{(gx)} \frac{\partial I_{aerr}}{\partial x} - I_x^{(gy)} \frac{\partial I_{aerr}}{\partial y}$$
$$\rho_v = \left(I_y^{(g)} - I_{xy}^{(gx)} - I_{yy}^{(gy)} \right) I_{aerr} - I_y^{(gx)} \frac{\partial I_{aerr}}{\partial x} - I_y^{(gy)} \frac{\partial I_{aerr}}{\partial y} \tag{4.2}$$

Now we have to break the derivative of I_{aerr}. But I_{aerr} contains seven terms and taking the derivative will create thirteen terms. Since we have to take the derivative with respect to x and y we end up with 26 terms. We have to do this for both ρ_u and ρ_v which leads to 52 terms (a few of them are the same but not many). Plus a couple of terms we already have! This is something that nobody wants to see printed.

The way out, is to introduce some higher level notation† that will allow us to write it in a concise form. We introduce the vectors of coefficients C_u and C_v every element of which is an image derivative and the vectors of unknowns F_u and F_v whose elements are the components of flow and their derivatives. Then we rewrite I_{aerr}

$$I_{aerr} = C_u \cdot F_u + C_v F_v + I_t$$

where

$$C_u = \begin{bmatrix} I_x^{(g)} \\ I_x^{(gx)} \\ I_x^{(gy)} \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad C_v = \begin{bmatrix} I_y^{(g)} \\ I_y^{(gx)} \\ I_y^{(gy)} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

and

†The notation is influenced by the syntax of MediaMath, an object oriented system for mathematical calculations related to multimedia

$$F_u = \begin{bmatrix} u \\ u_x \\ u_y \\ u_{xx} \\ u_{xy} \\ u_{yy} \end{bmatrix}, \quad F_v = \begin{bmatrix} v \\ v_x \\ v_y \\ v_{xx} \\ v_{xy} \\ v_{yy} \end{bmatrix}$$

and the $\cdot$ denotes dot product. Then

$$\frac{\partial I_{aerr}}{\partial x} = C_{u,x} \cdot F_u + R\_x(C_u) \cdot F_u + C_{v,x} \cdot F_v + R\_x(C_v) F_v + I_{tx}^{(g)}$$
$$\frac{\partial I_{aerr}}{\partial y} = C_{u,y} \cdot F_u + R\_y(C_u) \cdot F_u + C_{v,y} \cdot F_v + R\_y(C_v) F_v + I_{ty}^{(g)}$$

where $C_{u,x}$ is a vector every element of which is the derivative with respect to x of the corresponding element of C_u. Also $R\_x(C_u) \cdot F_u$ is equal to $C_u \cdot \frac{\partial F_u}{\partial x}$ but instead of taking the derivative of F_u, which is a mere rearrangement of its elements, we rearrange C_u. The rearrangement is done by the function R_x. So, now we can write Eq. (4.2) as

$$\begin{aligned} \rho_u &= C_{uu} \cdot F_u + C_{uv} \cdot F_v + C_{ut} \\ \rho_v &= C_{vu} \cdot F_u + C_{vv} \cdot F_v + C_{vt} \end{aligned} \tag{4.3}$$

where

$$C_{uu} = \left(I_x^{(g)} - I_{xx}^{(gx)} - I_{xy}^{(gy)} \right) C_u - I_x^{(gx)} \left(C_{u,x} + R\_x(C_u) \right) - I_x^{(gy)} \left(C_{u,y} + R\_y(C_u) \right)$$

$$C_{uv} = \left(I_x^{(g)} - I_{xx}^{(gx)} - I_{xy}^{(gy)} \right) C_v - I_x^{(gx)} \left(C_{v,x} + R\_x(C_v) \right) - I_x^{(gy)} \left(C_{v,y} + R\_y(C_v) \right)$$

$$C_{vu} = \left(I_y^{(g)} - I_{xy}^{(gx)} - I_{yy}^{(gy)} \right) C_u - I_y^{(gx)} \left(C_{u,x} + R\_x(C_u) \right) - I_y^{(gy)} \left(C_{u,y} + R\_y(C_u) \right)$$

$$C_{vv} = \left(I_y^{(g)} - I_{xy}^{(gx)} - I_{yy}^{(gy)} \right) C_v - I_y^{(gx)} \left(C_{v,x} + R\_x(C_v) \right) - I_y^{(gy)} \left(C_{v,y} + R\_y(C_v) \right)$$

$$C_{ut} = \left(I_x^{(g)} - I_{xx}^{(gx)} - I_{xy}^{(gy)} \right) I_t^{(g)} - I_x^{(gx)} I_{tx}^{(g)} - I_x^{(gy)} I_{ty}^{(g)}$$

$$C_{vt} = \left(I_y^{(g)} - I_{xy}^{(gx)} - I_{yy}^{(gy)} \right) I_t^{(g)} - I_y^{(gx)} I_{tx}^{(g)} - I_y^{(gy)} I_{ty}^{(g)}$$

A very important advantage of this formulation is that we completely factor out the coefficients from the unknowns. This means that if we use more than one filter *g* then we simply add the coefficients of each filter. Even better, if we want to introduce ideas from other approaches then we just formulate them in the same way and add the coefficients. In particular Nagel's algorithm and Horn and Schunck's algorithm can be very easily cast in this form and with very little overhead we can use them. Another idea that comes almost free, is Lucas and Kanade's. We can assume that the affine coefficients do not change much in a small window or equivalently, use every filter more than once, each time slightly shifted with respect to the others.

Another advantage is that when we solve the linear system we deal only with C_{uu}, C_{uv} etc that are arrays of images and the derivatives of the flow (which have to be recomputed every time). The whole iteration can be done using only operations on images like multiplication, addition, convolution (all of them separable) so that it is easily adaptable to SIMD type multiprocessors.

4.1. Selection of filters

Although the above development does not restrict the choice of filters, we should use ones that are inexpensive to compute with and make sense from a Computer Vision point of view. Some of their properties are

(1) Separable or semiseparable (e.g. their convolutions can be broken in a series of separable convolutions) for efficiency.

(2) It is easy to compute their derivatives in both space (to avoid direct computation of the derivative of the image) and frequency (to compute convolutions with gx).

(3) They have good localization in space (to avoid going outside areas that the affine approximation is adequate, and be able to work near discontinuities).

(4) All of the filters together cover large part of the spectrum. Otherwise we might leave information unused.

There are several commonly used filters that have these properties: Gaussian, Laplacian of Gaussian, Gabor etc. Filters that do not satisfy these properties are the sampling function (its derivative in frequency is discontinuous and if we multiply it with a Gaussian the later dominates) and a sine times a uniform function [24] because it is difficult to compute the second derivative of such a filter.

The experiments for this paper were done using a set of four Gaussians. We also use a smoothing term, which can be incorporated by adding λ to all the coefficients of second derivatives. We incorporate Lucas and Kanade's idea by convolving all the coefficients with a small Gaussian.

4.2. 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 case the properties we are looking for are

(1) Reasonable convergence rate. We are not very strict with this condition because the linear system turns out to be fairly stable numerically. The ill posedness and the instability come from the physical problem itself, not from its numerical formulation.

(2) We should be able to express the algorithm in terms of few simple image operations. This is very restrictive condition but otherwise the algorithm would not be suitable for implementation on a parallel SIMD architecture. This leaves only Jacobi iteration and Conjugate Gradient as candidates.

(3) Every cycle of the iteration must be inexpensive in CPU and memory since we work on images that are fairly big data structures.

The only candidates left are Jacobi iteration and Conjugate Gradient. The Jacobi has the advantage of simplicity and cheap cycle. The Conjugate Gradient has the advantage of fast convergence. An additional advantage for the later is that it does not diverge in the parts of the image that have little information.

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 concolved 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

5. Experiments

The experiments were done using both synthetic and real images. The filters used were Gaussians and derivatives of Gaussians. They performed more or less the same. The linear system solver used was Jacobi iteration with maximum of 60 iterations. The execution time ranged between 20 seconds for 64×64 images to 7 minutes for 150×150 images on a Sparc-10 using MediaMath.

5.1. Synthetic images

The sysnthetic image we used consists of two cosines in polar coordinates that rotate. The interframe rotation is 5 degrees.

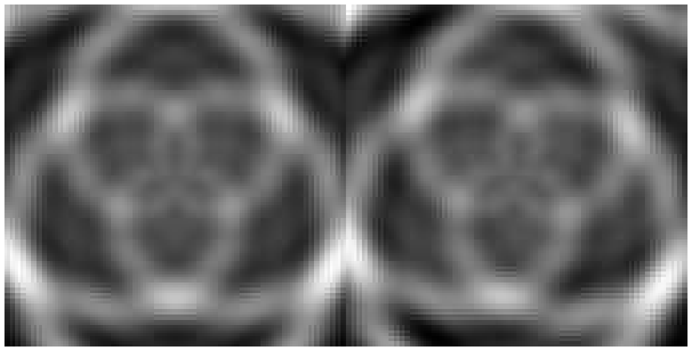

Figure 5.1 The synthetic image pair.

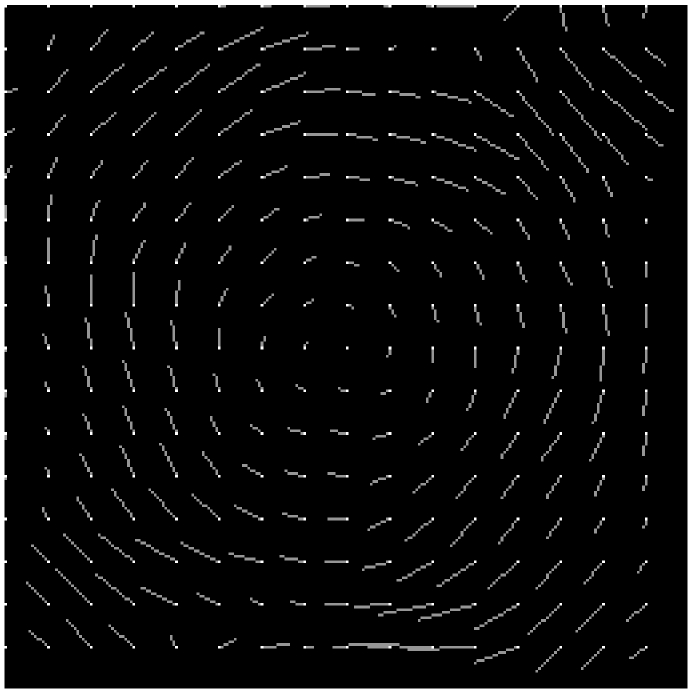
Figure 5.2 The flow field of the synthetic images.

5.2. Real images

The flow field real images was filtered to reject the areas of low confidence.

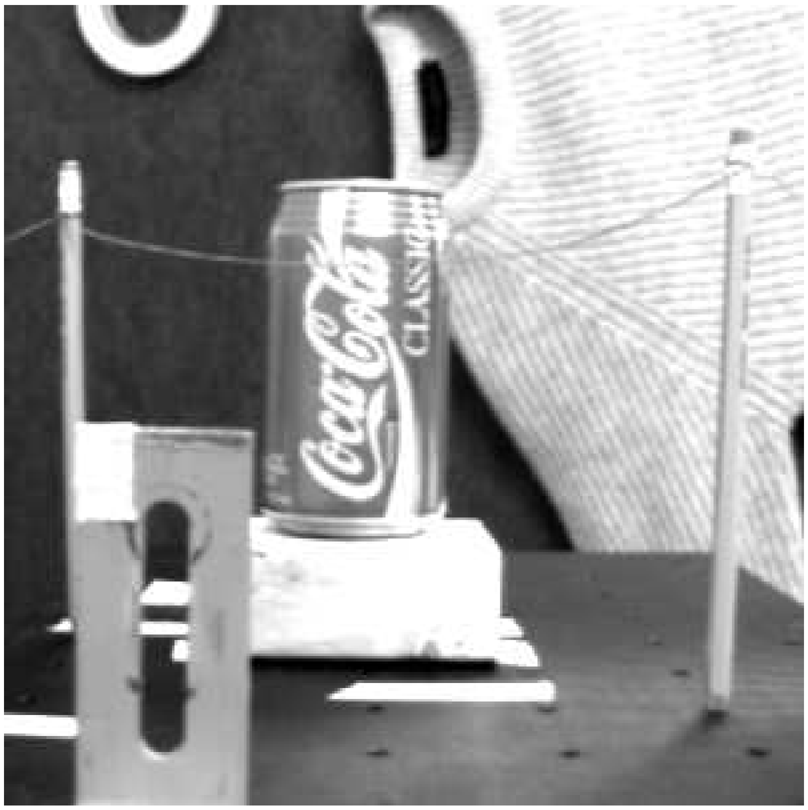

Figure 5.3 The first of the NASA sequence pair. The first and third images were used for the estimation of the flow.

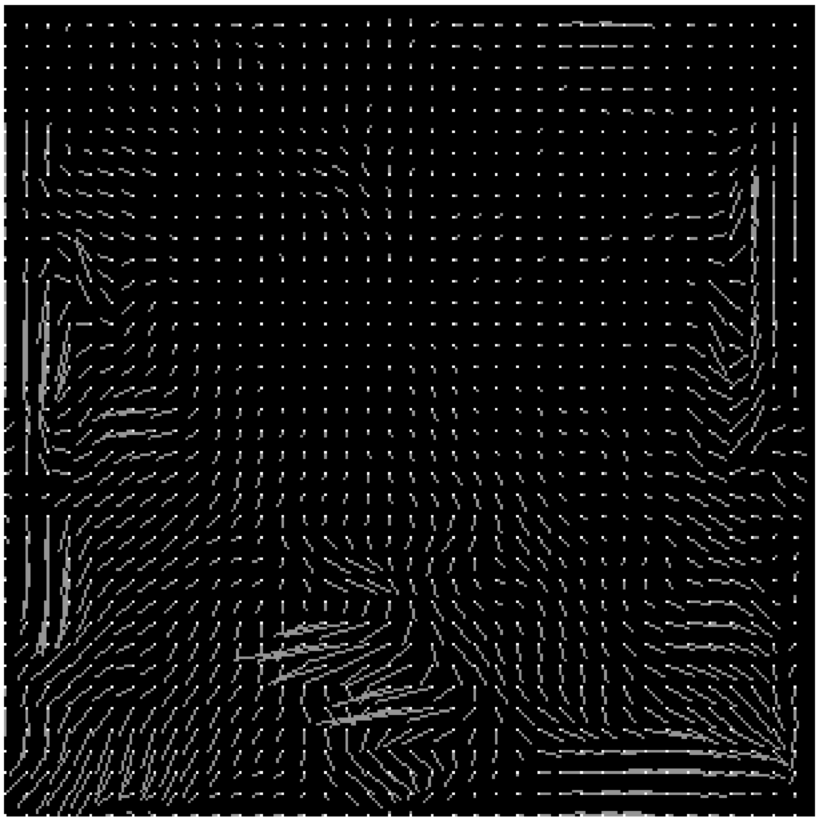

Figure 5.4 The flow field of the NASA sequence. The flow field is shown only in areas were the confidence measure (the determinant of the 2×2 diagonal blocks) is high.

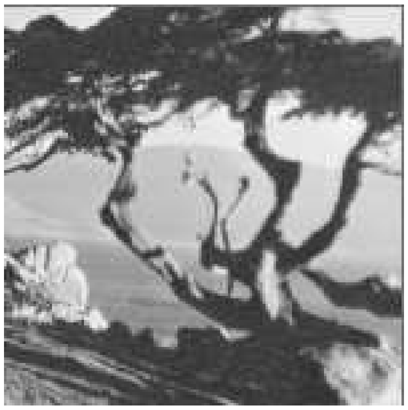

Figure 5.5 The first images of the diverging tree sequence. Images 10 and 12 were used.

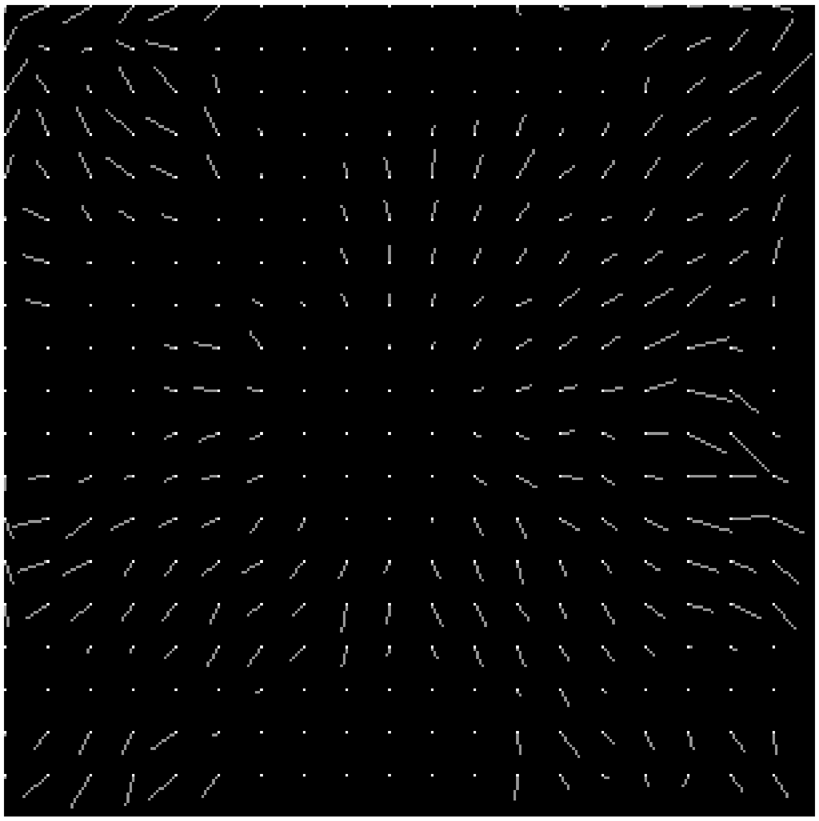

Figure 5.5 The flow field of the diverging tree sequence. The flow field is shown only in areas were the confidence measure (the determinant of the 2×2 diagonal blocks) is high.

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.

A very interesting development would be to incorporate phase information in this formulation. This is expected to improve the accuracy of the method, but, if done properly it should result in an formalism that unifies the gradient based and the phase based approaches.

7. Credits

The mathematical derivations for this paper were done with Maple. The plotting was done by Gnuplot. The implementation of this algorithm was done using MediaMath. Although the algorithm has fairly complicated structure, the implementation took only four 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.

Financial support was provided by NSERC.

References

1. Y. Aloimonos and Z. Duric, “Active egomotion estimation: A qualitative approach,” *ECCV*, (497-510).

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

3. 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 (Jul2 1993).

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

5. C. Fermuller, "Global 3-D Motion Estimation," *CVPR,* pp. 415-421 (1993).

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

7. 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).

8. 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).

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

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

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

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

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

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

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

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

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

18. S. Negahdaripour and B.K.P. Horn, “Determinig 3-D motion of planar objects from image brightness patterns,” *Proceedings of the IJCAI* **2** pp. 898-901 (August 1985).

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

20. M. E. Spetsakis, “Spectrum selective techniques for flow estimation,” *IAPR (submitted)*, (1994).

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

22. D. Weinshall, “Direct computation of Qualitative 3D shape and motion invariants,” *ICCV*, pp. 230-237 (1990).

23. D. Weinshall, “Qualitative depth from stereo, with applications,” *CVGIP* **49**(1990).

24. J. Weng, “Image matching using the windowed Fourier phase,” *IJCV* **11(3)** pp. 211-236 (1993).