FLUID Image analysis and Description (FLUID)
Project No. FP6-513663
FET - Open Domain, Priority IST

Deliverable 2.1

Report on Filter Bank Design for Local Fluid Motion Estimation

Jing Yuan, Florian Becker, Christoph Schnörr
CVGPR group, University of Mannheim

Due date of deliverable: 30 November 2005
Actual submission date: 30 November 2005
Start date of project: 1 December 2004
Duration: 3 years

Project cofunded by the European Commission within the Sixth Framework Programme (2002-2006).

Dissemination level: PU (Public)
## Contents

1 Introduction .......................................................... 1  
1.1 Design Requirements .............................................. 1  
1.2 Local Filters and Motion Estimation: A Brief Overview ....... 2  
1.3 Design Decisions .................................................. 4  

2 Multiscale Local Motion Estimation .............................. 5  
2.1 Approach .......................................................... 5  
2.2 Filter Design ...................................................... 6  
  2.2.1 Low-pass filters ............................................. 7  
  2.2.2 Derivative Filters ........................................... 7  
2.3 Image Pyramids .................................................. 8  
2.4 Interpolation with Splines ....................................... 9  
  2.4.1 Uniform B-Splines ......................................... 9  
  2.4.2 Computing the Coefficients ................................. 10  
  2.4.3 Image Interpolation ....................................... 12  
2.5 Multiscale Motion Estimation ................................... 13  

3 From Local to Global Motion Estimation ....................... 16  
3.1 Ill-Posedness of Local Motion Estimation ..................... 16  
3.2 Regularization ................................................... 17  
  3.2.1 Numerical Rank and Truncation ............................ 17  
  3.2.2 Filter Factors ................................................ 17  
3.3 Global Variational Motion Estimation ........................ 18  
  3.3.1 Data Term .................................................... 18  
  3.3.2 Global Regularization ..................................... 20
Chapter 1

Introduction

This chapter gives a brief overview over local filter design in connection with local image motion estimation (section 1.2). The overview is based on the requirements that we impose on the design of a multiscale stage for local motion estimation. This requirements are specified and discussed in (section 1.1).

The stage itself and the design of corresponding filters are described in section 2. Next, we specify how it can be used as part of any non-local variational approach to motion estimation within the FLUID project (section 3).

This report accompanies a second report [BYS05] which describes all technical details of a corresponding implementation (demonstrator). The implementation is equipped with an interface allowing for its use as part of non-local variational approaches to motion estimation, to be developed within the overall project.

1.1 Design Requirements

Local motion estimation is concerned with estimation the apparent image velocity \( u(x,t) \) at some image position \( x \in \Omega \) and some point of time \( t \in [0,T] \). Depending on the local image structure, it is possible to recover the full flow vector, or only a component of it (e.g., the so-called normal flow). Moreover, in homogeneous image regions, no motion can be estimated at all. This problem is known as the aperture problem.

Target of the design of a computational stage for local motion estimation are non-local variational approaches of the form

\[
  u^* = \arg\min_{u \in U} J(u), \quad J(u) = D(u) + R(u) \tag{1.1}
\]
The stage is incorporated into the so-called data-term \( D(u) \) which is complemented by a regularizing functional \( R(u) \). We refer to [HS81, WS01] for a prototypical example and a survey of extensions.

In connection with (1.1), the following requirements arise for the design of a computational stage for local motion:

(i) Motion estimation should operate on multiple scales. 'Multiscale' refers to the capability to deal with large motions leading to aliasing along the time axis at the original image resolution.

(ii) Motion estimation should be local. 'Local' refers to small spatial support: As part of a non-local variational approach, using a large spatial supports is not reasonable because the corresponding unspecific smoothing effect interferes in a rather uncontrolled way with the effect of variational regularization terms, which are a major topic of the investigations within this project.

(iii) Motion estimation should provide a confidence measure allowing for proper mathematical models of the interplay in (1.1) between local motion estimation and non-local variational regularization.

The subsequent overview over local filters for motion estimation focuses mainly on the requirements (i) and (ii) above. Issue (iii) is dealt with in section 3.

1.2 Local Filters and Motion Estimation: A Brief Overview

Filter design for local motion estimation has a long history in image processing and computer vision. Whereas the filters used in the seminal paper by Horn and Schunck [HS81] were rather crude from the viewpoint of signal processing, closer investigations of the subject were published rather early.

Hashimoto and Sklansky [HS87], for example, presented a thorough study of Gaussian derivative filtering, taking into account aspects of discretization and efficient implementation. Using Krawtchouk polynomials as discrete analogs of Hermite polynomials, which are orthogonal with respect to the Gaussian-weighted \( L_2 \) inner product, they exhibited some optimal properties of the resulting filters, such as maximal energy concentration within the Nyquist frequency band for filters with small spatial support. Moreover, they enjoy the following favourable properties:
- They are robust in terms of (i) approximating the ideal frequency response \((i\omega)^k\) of the \(k\)-th derivative for small frequencies, and (ii) having low-pass characteristics for noise suppression at higher frequencies. Indeed, the latter decay is exponential and hence very fast.

While it is straightforward to develop impulse responses providing good pointwise approximations within \(\omega \in [-\omega_g, \omega_g]\) of the ideal impulse response for larger \(\omega_g\), this is always at the cost of a considerably larger sensitivity to noise.

- They are computationally efficient due to separability, and because all filter coefficients are simple rational numbers, allowing for special implementations with integer arithmetic, if required.

Major attempts to improve filters for local motion estimation include studies of their invariance properties \([VF92, DV94, FS97]\). For example, computation of the gradient should not depend on the orientation of the local coordinate system. While marginal improvements are possible, the resulting coefficients are on longer rational numbers. This seems to be a high price compared to the good rotational invariance already achieved with the filter design in \([HS87]\).

A second line of research concerns local motion computation using analytical bandpass filters. Starting point was work by Fleet and Jepson \([FJ90]\) who showed the increased robustness of using the phase of such filters instead of image intensity. On the other hand, locally constant motion over a \(15 \times 15 \times 15\) pixel spatio-temporal support was assumed, and more than 20 filters were used for sampling the frequency space, in order to determine the orientation of the frequency plane which corresponds to the translation of local image patches.

To alleviate these computational costs and to reduce the filter support, wavelet-based versions were investigated \([SFAH92, MK98, Kin01, Ber01]\). Using standard dyadic multiresolution schemes \([Mal98]\) causes a severe loss of invariance, however. Therefore, by introducing redundant signal representations, attempts were made to optimize the compromise between approximate invariance (translation, rotation) and computational efficiency \([MK98, Kin01]\).

Our own investigation of these schemes \([Neu04, chapter 3]\), in connection with invariant feature extraction for pattern recognition, showed however, that the approximation to rotational invariance is not satisfying. While these deficiencies can be improved \([NS05]\), it leads again to elaborate filter banks represented in frequency space that do not allow for efficient implementations in the spatial or spatio-temporal domain.
1.3 Design Decisions

All partners agreed on a first project meeting that it would be worthwhile to design and to implement a first version of a standard data term $D(u)$, to be used by any partner as part of variational approaches of the form (1.1). This will allow in subsequent project phases for the comparability of different variational approaches to motion estimation.

Based on

- the discussion in the previous section,
- our experience with optical flow computation in computer vision (e.g., [BWS05] and references),
- and our recent experience with applying a suitable modification of the prototypical variational approach of Horn and Schunck [HS81] to Particle Image Velocimetry [RKNS05],

we decided to design a data term $D(u)$ as a first standard, by modifying the well-known Lukas-Kanade approach [LK81, BFB94, BM04] in view of its incorporation into (1.1) and the corresponding requirements discussed in section 1.1.
Chapter 2

Multiscale Local Motion Estimation

In this section, we describe

- the local approach to motion estimation,
- the corresponding filter design,
- the scheme used for image interpolation and warping, and
- the multiscale version of the motion estimation approach.

The incorporation of the approach into a non-local variational approach will be exemplified in section 3.

2.1 Approach

Let

\[ G_\sigma(x-y) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp \left( -\frac{1}{2\sigma^2} \|x-y\|^2 \right), \quad d = 2 \text{ or } 3 \]

denote a Gaussian located at \( x \) in the image domain. Assuming constant image motion within a region size related to the parameter \( \sigma \), the approach consists in minimizing

\[ E(u) = \int G_\sigma(x-y)\|\nabla g^\top u + g_t\|^2 \, dy \quad (2.1) \]
Here, \( g(x,t) \) denotes the image sequence function with gradient and partial derivatives

\[
\nabla g = \left( \frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2} \right) g, \quad g_t = \frac{\partial}{\partial t} g
\]

Furthermore, \( g(x,t) \) is assumed not to change during motion, leading to the common constraint equation \([HS81]\)

\[
\nabla g^\top u + g_t = 0
\]  

(2.2)

Expanding the integrand of (2.1) and computing the derivative with respect to \( u \) leads to the linear system

\[
Au = b
\]  

(2.3)

where

\[
A = G_\sigma \ast [(\nabla g)(\nabla g)^\top]
\]

(2.4a)

\[
b = -G_\sigma \ast (g_t \nabla g)
\]

(2.4b)

and convolution \( \ast \) is applied element-wise. Matrix \( A \) is known as structure tensor in the literature.

Note: Both \( G_\sigma \) and the partial derivatives \( \nabla g, g_t \) in (2.3) and (2.4) are implemented with the filters described in the next section.

### 2.2 Filter Design

We describe a class of filters with the following properties:

- The filters are separable, that is multi-dimensional filters are formally constructed by taking the outer product of one-dimensional filters. For example, if \( h^0 \) denotes the impulse response of a low-pass filter (zero'th-derivative and noise suppression), and \( h^1 \) the impulse response of a first-derivative filter, then the partial derivative with respect to \( x_2 \) of an image sequence \( g(x,t) \) is estimated:

\[
\frac{\partial}{\partial x_2} g(x_1, x_2, t) \approx (h^0 \otimes h^1 \otimes h^0) \ast g
\]

Of course, the implementation of the right-hand side is done by subsequent one-dimensional convolutions.

As a result, it suffices in the following to consider the design of one-dimensional filters.
The filters are “multiscale” in the sense that a larger scale can be computed from a smaller scale by “adding” the scale difference, similarly to the semigroup property in the linear continuous scale space.

All filter coefficients are simple rational numbers that in principle can be efficiently implemented on dedicated hardware.

Maximal noise suppression through exponential decay as $\omega \rightarrow \pi$.

### 2.2.1 Low-pass Filters

Let $3 \leq n \in \mathbb{N}$, $n$ odd, denote the size of the (discrete) impulse response. Then the definition is:

$$h^0_n(x) = \frac{1}{2^{n-1}} \binom{n-1}{x}, \quad x = 0, \ldots, n-1 \quad (2.5)$$

Convolution approximates Gaussian smoothing with parameters $\mu = \frac{n-1}{2}$ and $\sigma^2 = \frac{n-1}{4}$. Figure 2.1 below demonstrates why this filter is preferable instead of sampling the Gaussian, besides its efficient implementation.

![Frequency responses](image)

Figure 2.1: Frequency responses $\hat{h}^0_n(\omega)$ for $n = 3, 5, 11$ (blue), and for the sampled Gaussian low-pass (black). For filters with small spatial support, the energy of the approximate filter (blue curves) is better contained within the Nyquist interval.

### 2.2.2 Derivative Filters

Computation of derivatives by convolution is based on the following representation through the Fourier transform:

$$\frac{d}{dx}(h^0 \ast g(x)) \iff (i\omega)(\hat{h}^0(\omega)\hat{g}(\omega)) = ((i\omega)\hat{h}^0(\omega))\hat{g}(\omega)$$

and

$$\frac{d^k}{dx^k}(h^0 \ast g(x)) = \frac{d}{dx}\left(\frac{d^{k-1}}{dx^{k-1}}(h^0 \ast g(x))\right)$$
These equations say that derivative estimation, in conjunction with noise suppression, amounts to convolve the data with the derivative of a low-pass filter.

The following modification of the definition of low-pass filters (2.5) takes this into account:

$$h_n^k(x) = \frac{1}{2^{n-k-1}} \sum_{i=0}^{k} (-1)^i \binom{n-k-1}{x-i} \binom{k}{i}, \quad x = 0, \ldots, n-1 \quad (2.6)$$

The implementation is very simple:

(i) Convolve $n - k - 1$ times with the impulse response $(1, 1)$.

(ii) Convolve $k$ times with the impulse response $(1, -1)$.

(iii) Normalize.

While it is straightforward to design filters with excellent pointwise approximation of the ideal derivative filter, this also results in poor noise suppression – see figure 2.2. Therefore, when working with real image data, (2.6) is our choice.

![Figure 2.2: Frequency responses of two classes of derivative filters in comparison with the ideal derivative filter (blue line). Left: Approximation of Gaussian derivative filters (2.6). The low-pass band shrinks for increasing size $n$ of the impulse response, due to larger noise suppression. Right: Another class of derivative filters which much better approximates the ideal filter the larger is $n$. Noise suppression is poor, however. For real image data, therefore, the filter class shown on the left is preferable.](image)

### 2.3 Image Pyramids

For each point of time $t$, a given image $g(x) = g(x, t)$ is transformed into a so-called image pyramid by low-pass filtering and subsampling. Proper
filtering depends on the sampling theorem and on the distribution of the spectral energy $|\hat{g}(\omega)|$.

Figure 2.3 illustrates this process. More details relevant for motion estimation are given in section 2.5.

Figure 2.3: From left to right: Image pyramid by low-pass filtering and subsampling. Top: Improper filtering leads to aliasing. Bottom: Proper filtering does not create spurious structures.

2.4 Interpolation with Splines

Image motion leads to transformations of functions which do not conform to the image grid. Consequently, interpolation is needed for transferring arbitrary functions to and from image grids.

Since several decades [Pra78] cubic splines are known to accurately approximate the ideal interpolation function $\text{sinc}(x)$, while being compactly supported. We describe next a corresponding efficient interpolation scheme based on the work of Unser [Uns99, UAE93a, UAE93b].

We consider again the one-dimensional case first, and then consider image interpolation.

2.4.1 Uniform B-Splines

Splines are smooth piecewise-polynomial functions. A spline of degree $n$ is $n - 1$ times continuously differentiable. As a result, $n$ degrees of freedom of
each polynomial piece are already fixed through these continuity constraints, leaving just a single degree of freedom.

In the 1-D case and for a uniform grid, the $n$-degree polynomial spline is defined as

$$f(x) = \sum_{k \in \mathbb{Z}} c(k) \beta^n(x - k)$$

(2.7)

where $\beta^n(x)$ denotes the uniform B-spline basis functions

$$\beta^n = \beta^0 \ast \ldots \ast \beta^0(x), \quad \beta^0 = \begin{cases} 1, & -\frac{1}{2} < x < \frac{1}{2} \\ \frac{1}{2}, & |x| = \frac{1}{2} \\ 0, & \text{otherwise} \end{cases}$$

(2.8)

The commonly used cubic B-spline ($n = 3$)

$$\beta^3(x) = \begin{cases} \frac{2}{3} - \frac{|x|^2}{2}, & 0 < |x| < 1 \\ \frac{1}{2} \left(1 - |x| \right)^3, & 1 < |x| < 2 \\ 0, & |x| \geq 2 \end{cases}$$

(2.9)

performs high-quality interpolation at a low computational cost. Of course, it is possible to use a higher-order spline interpolation. The accuracy does not benefit much, however, in comparison to the increased computational costs.

Due to (2.7), each spline function is represented by the coefficients sequence $\{c(k)\}$. To perform interpolation, the main work is to compute these coefficients. This will be addressed in the next section.

Once the coefficients have been computed, it is easy to sample arbitrary function values $f(x)$, or to find the derivative or integral of the function $f$ based on corresponding operations applied to the basis function $\beta^n(x)$:

$$\frac{d\beta^n(x)}{dx} = \beta^{n-1}(x + 1/2) - \beta^{n-1}(x - 1/2);$$

(2.10)

$$\int_{-\infty}^{x} \beta^n(x) \, dx = \sum_{k=0}^{+\infty} \beta^{n+1}(x - 1/2 - k).$$

(2.11)

### 2.4.2 Computing the Coefficients

Given the samples $f(k)$, the interpolation problem based on (2.7) amounts to find the spline coefficients from the conditions

$$\sum_{l \in \mathbb{Z}} c(l) \beta^n(x - l)|_{x=k} = f(k).$$

(2.12)
This can be rewritten as a convolution
\[ f(k) = (b^n * c)(k) \]  
where \( b^n = \beta^n(k - l) \) are filter coefficients. Then the spline coefficients \( c(k) \) can be found by inverse filtering [UAE93b]
\[ c(k) = (b^n)^{-1} * f(k) \]  
The convolution kernel \((b^n)^{-1}\) can be easily computed by the z-tranform of \( b^n \), and be implemented efficiently by a cascade of first-order causal and anti-causal recursive filters.

For example, the z-tranform of the cubic B-spline \( b^3 \) from (2.9) is
\[ B^3(z) = \frac{z + 4 + z^{-1}}{6} \]
Thus, the corresponding z-transform of the filter \((b^3)^{-1}\) is
\[ (b^3)^{-1}(k) \leftrightarrow \frac{6}{z + 4 + z^{-1}} \]
The right-hand transform leads to the following causal and anti-causal recursive algorithm [Uns99]:
\[ c^+(k) = f(k) + z_1 c^+(k - 1), \quad k = 1, \ldots, N - 1 \]  
\[ c^-(k) = z_1 (c^-(k + 1) - c^+(k)), \quad k = N - 2, \ldots, 0 \]
where \( z_1 = -2 + \sqrt{3} \) and \( c^-(k) = c(k)/6 \). Assuming the mirror-symmetric boundary conditions, the two initializing value \( c^+(0) \) and \( c^-(N - 1) \) are defined as
\[ c^+(0) = \frac{1}{1 - z_1^{2N-2}} \sum_{k=0}^{2N-3} f(k) z_1^k \]  
\[ c^-(N - 1) = \frac{z_1}{1 - z_1^2} \left( c^+(N - 1) + z_1 c^+(N - 2) \right) \]
The algorithm is numerically stable and fast. Interpolation can then be done by a simple convolution. Since each basis function is compactly supported, this scheme provides a very good compromise between accuracy and computational complexity [Uns99].

The spline-based interpolation (2.7) generalizes often-used interpolation methods of the form
\[ f(x) = \sum_{k \in \mathbb{Z}} f_k \psi(x - k) \]
by using an extra series of coefficients \( \{c_k\} \), providing additional flexibility so as to construct reliable and fast interpolation schemes with compact support [TBU00]. Conversely, as the \( c_k \) coefficients are still related to the samples \( f_k \), spline-based interpolation may still be regarded as a normal interpolation method where the coefficients are defined implicitly.

In the following section, we compare the spline-based interpolation with two other common interpolation methods: Bilinear interpolation and bicubic interpolation.

### 2.4.3 Image Interpolation

For the 2-D case and a uniform grid, the spline function reads

\[
f(x, y) = \sum_{k=k_1}^{(k_1+K-l)} \sum_{l=l_1}^{(l_1+K-l)} c(k, l) \beta^n(x - k) \beta^n(y - l) \tag{2.19}
\]

with \( k_1 = k_1(x) = \left[ x - (n + 1)/2 \right] \), \( l_1 = l_1(y) = \left[ y - (n + 1)/2 \right] \)

and \( K = \text{support}\{\beta^n\} = n + 1 \)

The parameters \( c(k, l) \) are constructed by applying 1-D filtering (2.14) successively along \( x \) and \( y \) axis, respectively. It is obvious that when the spline coefficients \( c(k, l) \) are given, the interpolated value at position \((x, y)\) can be computed by the sum within a compactly supported area around \((x, y)\). In this work, the spline-based interpolation is used to implement image transformation (warping) and image resampling, i.e. the rescaling procedures between the finer scale and the coarser scale.

In order to show the main advantages of spline-based interpolation for image warping, we use a highly relevant geometric transformation, rotation, which is successively applied \( k \) times to a given image. Figures (2.4) show the difference between spline-based image interpolation and bilinear interpolation and bicubic interpolation, respectively (both latter methods are implemented as function \texttt{interp2} in Matlab). It can be clearly seen that spline-based interpolation generates much smaller errors than other two interpolation methods with much less running time. Although the bicubic interpolation is more accurate than the bilinear method, it needs more running time to reach the higher accuracy.

In the algorithm for multiscale motion estimation, resampling procedures are necessary to construct 2-D functions at some finer level from a coarser level, or at some coarser level from a finer level with some zooming factor \( k \).
According to (2.19), the spline-based resampling is implemented by selecting the coordinate displacements \((i + 1/k, \ j + 1/k)\), \(i = 1, \ldots, m\), \(j = 1, \ldots, n\), where \(k\) is the rescaling factor.

The comparison between spline-based resampling and bilinear resampling is shown in figure (2.5). The close-up views clearly exhibit the very good performance of the spline-based method, whereas the bilinear-based method generates some visible artifacts.

### 2.5 Multiscale Motion Estimation

We summarize the overall algorithm for multiscale motion estimation. A detailed description of all functions is given in the second part of this report [BYS05].

**Algorithm:** Multiscale optical flow estimation

**Input:** image pair \(g_1, g_2\)

**Output:** estimated warp \(u^0\)

\[
g_1^{(0)}, \ldots, g_1^{(d_{\text{coarsest})}} \leftarrow \text{multiscaleFilter2}(g_1);
g_2^{(0)}, \ldots, g_2^{(d_{\text{coarsest})}} \leftarrow \text{multiscaleFilter2}(g_2);
\]

\(u^{(d_{\text{coarsest})}} \leftarrow \text{Identity} ;\)

**for** \(i\) **from** level \(d_{\text{coarsest}}\) **down to** 0 **do**

**repeat**

\[
\begin{align*}
\bar{g}_1 &\leftarrow \text{warpImage}(g_1^{(i)}, +\frac{1}{2}u^{(i)}) ; \\
\bar{g}_2 &\leftarrow \text{warpImage}(g_2^{(i)}, -\frac{1}{2}u^{(i)}) ; \\
\Delta u &\leftarrow \text{estimateWarp}\left(\bar{g}_1, \bar{g}_2\right) ;
\end{align*}
\]

**if** \(\Delta u\) **improves some error measurement** **then**

\[
u^{(i)} \leftarrow u^{(i)} + \Delta u ;
\]

**end**

**until** some stopping criterion is met;

**if** \(i > 0\) **then**

\[
u^{(i-1)} \leftarrow \text{rescaleWarp}(u^{(i)}, g^{(i-1)}) ;
\]

**end**

**end**

**Algorithm 1:** Algorithmic description of a multiscale optical flow estimator.
Figure 2.4: The **upper left image** is an original 2-D PIV image downloaded from the website of The Visualization Society of Japan: http://piv.vsj.or.jp/piv/image-e.html. This image is rotated by 360 degree in 23 steps with different interpolation methods. The **upper right image** shows the result using spline-based image interpolation. The **bottom left image** results from bicubic interpolation, and the **bottom right image** shows the result from bilinear interpolation. The detailed error measurement in the table below is based on the average and standard derivation of $\|g(x) - g_{rot}(x)\|_2$.

<table>
<thead>
<tr>
<th></th>
<th>err.</th>
<th>spline</th>
<th>bilinear</th>
<th>bicubic</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>3.62e-3</td>
<td>1.56e-2</td>
<td>7.25e-3</td>
<td></td>
</tr>
<tr>
<td>st. deviation</td>
<td>1.36e-2</td>
<td>4.84e-2</td>
<td>2.47e-2</td>
<td></td>
</tr>
<tr>
<td>max</td>
<td>0.36</td>
<td>0.70</td>
<td>0.49</td>
<td></td>
</tr>
<tr>
<td>run. time (sec.)</td>
<td>3.4</td>
<td>10.6</td>
<td>26.4</td>
<td></td>
</tr>
</tbody>
</table>
Figure 2.5: Top: original image. This image is zoomed 10-times by spline-based resampling, and by bilinear resampling. The respective two images are shown in the second row. Left: bilinear. Right: spline-based method. Corresponding close-up views in the third row show artifacts generated by the bilinear method.
Chapter 3

From Local to Global Motion Estimation

In this section, we first make the ill-posedness of the local estimation scheme (2.3) explicit. Based on this, we discuss regularizations as well as the embedding of the regularized local motion estimation into a global variational approach (1.1).

3.1 Ill-Posedness of Local Motion Estimation

Definition (2.4) shows that the structure tensor $A$ is symmetric and positive definite. Its spectral decomposition into eigenvectors $e_1, e_2$ and eigenvalues $\lambda_1, \lambda_2$ reads:

$$ A = (e_1 \ e_2) \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} (e_1 \ e_2)^\top, \quad \lambda_1 \geq \lambda_2 \geq 0 $$

Solvability of (2.3) depends on the rank of $A$, that is the number of non-vanishing eigenvalues $\lambda_1, \lambda_2$, which in turn depends on the local grayvalue structure defining $A$ in (2.4).

We distinguish the following cases:

**Rank = 2:** Both components of $u$ can be locally computed by solving (2.3).

**Rank = 1:** Image structure varies locally in a single direction only. As a result, $\lambda_2 = 0$, and only a single component of $u$, the so-called normal flow, can be computed. This is called the aperture problem.

**Rank = 0:** Both eigenvalues vanish $\lambda_{1,2} = 0$ due to a homogeneous image region. No motion information be computed.
Given the expansion of $u$ and $b$ in (2.3) in the corresponding coordinate system

\[ u = u^1 e_1 + u^2 e_2, \quad b = b^1 e_1 + b^2 e_2 \]

the solution to (2.3) can be expressed as:

\[ u_{loc} = \begin{cases} 
\frac{b^1}{\lambda_1^1} e_1 + \frac{b^2}{\lambda_1^2} e_2 , & \lambda_1 > 0 , \lambda_2 > 0 \\
\frac{b^1}{\lambda_1} e_1 , & \lambda_1 > 0 , \lambda_2 = 0 \\
0 , & \lambda_1 = \lambda_2 = 0 
\end{cases} \quad (3.1) \]

Unfortunately, it may quite often happen in practice, that both or one eigenvalue is very small, $\lambda_2 \ll 1$, providing a typical rank-deficient problem [Han98]. In this case, even when the data $b$ on the right-hand side in (2.3) are contaminated with small errors $\delta b$ only – which is unavoidable in practical situations – then the results can easily become corrupted and useless.

We describe next two regularization approaches to cope with such situations.

### 3.2 Regularization

#### 3.2.1 Numerical Rank and Truncation

The simplest way to deal with the ill-conditioned structure tensor $A$ in (2.3) is to define a suitable bound to determine the numerical rank. The latter is defined by two tolerance parameters $\epsilon$ and $\eta$ and leads to the following modification of (3.1):

\[ \tilde{u}_{loc} = \begin{cases} 
\frac{b^1}{\lambda_1^1} e_1 + \frac{b^2}{\lambda_2^2} e_2 , & \lambda_1^2 + \lambda_2^2 > \eta^2 , \frac{\lambda_2^2}{\lambda_1^1} > \epsilon \\
\frac{b^1}{\lambda_1} e_1 , & \lambda_1^2 + \lambda_2^2 > \eta^2 , \frac{\lambda_2^2}{\lambda_1^1} \leq \epsilon \\
0 , & \lambda_1^2 + \lambda_2^2 \leq \eta^2 
\end{cases} \quad (3.2) \]

Parameter $\eta$ is used to determine numerically the zero-rank case. Parameter $\epsilon$ measures the condition number of the matrix $A$ to distinguish between the rank-1 and rank-2 cases.

#### 3.2.2 Filter Factors

A disadvantage of the previous method is the extra computational cost of the spectral decomposition of $A$ decomposition. An alternative is to solve

\footnote{We use superscripts to distinguish the original coefficients, written with subscripts, related to the global image coordinate system.}
the least square problem

\[ \min_u \|Au - b\|^2 + \alpha \|u\|^2 \]  

(3.3)

It is easy to see that problem (3.3) leads to the system

\[ (A^\top A + \alpha I)u_{\text{reg}} = A^\top b \]  

(3.4)

Because the matrix \((A^\top A + \alpha I)\) has the same eigenvectors as \(A\) and both eigenvalues are always larger than zero, we do no longer have to distinguish explicitly the three cases related to the rank of \(A\).

Nevertheless, it is instructive to compare the solution of (3.4) with that of (3.2). The former reads

\[ u_{\text{reg}} = f_1 \frac{b_1}{\lambda_1} e_1 + f_2 \frac{b_2}{\lambda_2} e_2 \]  

(3.5)

where we call \(f_i, i = 1, 2\), the filter factors:

\[ f_i = \frac{\lambda_i^2}{\lambda_i^2 + \alpha}, \quad i = 1, 2 \]  

(3.6)

Clearly, when \(\lambda_i \gg 0\), we have \(f_i \rightarrow 1\). On the other hand, bad directions \(\lambda_i \rightarrow 0^+\) are filtered out by having \(f_i \rightarrow 0\). As the result, \(f_1, f_2\) can be considered as indicator functions which may be used for analyzing the structure tensor:

**Rank = 2:** \(f_1 \simeq 1\) and \(f_2 \simeq 1\)

**Rank = 1:** \(f_1 \simeq 1\) and \(f_2 \simeq 0\)

**Rank = 0:** \(f_1 \simeq 0\) and \(f_2 \simeq 0\)

Figure (3.1) shows an example illustrating the information provided by the filter factors \(f_{1,2}\).

### 3.3 Global Variational Motion Estimation

#### 3.3.1 Data Term

We address next the question of how to design a data term \(D(u)\) in (1.1), based on the previous considerations.
It is clear that reasonable data terms $D(u)$ have to account for the information actually provided by $u_{\text{loc}}$. According to (3.1), we make the ansatz:

$$D(u) = \int_{\Omega} \| f(u; A, u_{\text{loc}}) \|^2 dx$$

with

$$f(u; A, u_{\text{loc}}) = \begin{cases} u - u_{\text{loc}}, & \lambda_1 > 0, \lambda_2 > 0 \\ (u^\top e_1 e_1 - u_{\text{loc}}), & \lambda_1 > 0, \lambda_2 = 0 \\ 0, & \lambda_1 = \lambda_2 = 0 \end{cases} \quad (3.7)$$

At each location $x \in \Omega$ in the image domain, function $f(u; A, u_{\text{loc}})$ defined in (3.7) requires the global motion field to be compatible with the local measurements $u_{\text{loc}}$ only within the subspace $N(A)^\perp$, i.e.

$$f(u; A, u_{\text{loc}}) = P_{N(A)^\perp} u - u_{\text{loc}} \quad (3.8)$$

where $P_{N(A)^\perp}$ is the orthogonal projection operator to the complement onto the nullspace $N(A)$ of $A$ at location $x$:

$$P_{N(A)^\perp} = A^+ A = \begin{cases} e_1 e_1^\top + e_2 e_2^\top, & \lambda_1 > 0, \lambda_2 > 0 \\ e_1 e_1^\top, & \lambda_1 > 0, \lambda_2 = 0 \\ 0, & \lambda_1 = \lambda_2 = 0 \end{cases} \quad (3.9)$$

Like in the previous section, we wish to avoid the explicit distinction of these three cases. In view of (3.4), this leads us to the regularized version of (3.8)

$$f(u; A, u_{\text{reg}}) = A^+_\alpha A - A_{\text{reg}} = A^+\alpha (Au - b) \quad (3.10)$$

and the corresponding data term [Sch93]

$$D(u) = \int_{\Omega} \| A^+_\alpha (Au - b) \|^2 dx \quad (3.11)$$

where

$$A^+_\alpha = (A^\top A + \alpha I)^{-1} A^\top$$

Using the representation (3.5), we may rewrite integrand of the data term

$$\| f(u; A, u_{\text{reg}}) \|^2 = f_1^2(u^1 - u^1_{\text{reg}})^2 + f_2^2(u^2 - u^2_{\text{reg}})^2$$

which makes very explicit how confidence measures are incorporated through the filter factors defined in (3.6).
3.3.2 Global Regularization

Functional (1.1) with the data term as defined in (3.11) is well-posed for convex regularizers $R(u)$. This has been shown for quadratic regularizers in [Sch93], and for non-quadratic regularizers in [WS01].

**Proposition 1 ([Sch93, WS01])** Let $R(u)$ in (1.1) be convex with respect to $H^1(\Omega)^2$, and the functions $a_{11}, a_{12}, a_{22}$ defined by

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{pmatrix} := A^+_a A$$

be measurable and (almost everywhere) bounded. Suppose that $a_{11}, a_{12}$ and $a_{12}, a_{22}$, respectively, are linearly independent as elements of $L^2(\Omega)$. Then the functional (1.1) is strictly convex and has a unique global minimum $u$. 
Figure 3.1: The structure tensor $A$ is computed for the image shown on the top. The two columns with coloured images show the filter factors $f_1, f_2$, respectively, for different parameter values. Blue indicates $f_i = 0$, and red $f_i = 1$. 2nd row: $\alpha = 10^{-6}$, size of $G_{\sigma}$: $n = 15$. 3rd row: $\alpha = 10^{-8}$, size of $G_{\sigma}$: $n = 15$. 4th row: $\alpha = 10^{-6}$, size of $G_{\sigma}$: $n = 5$. 
Bibliography

[Ber01] C.P. Bernard. Discrete wavelet analysis for fast optic flow com-


