The B-Spline Channel Representation: Channel Algebra and Channel Based Diffusion Filtering

Report LiTH-ISY-R-2461

Michael Felsberg* Hanno Scharr† Per-Erik Forssén*

mfe@isy.liu.se hannox.scharr@intel.com perfo@isy.liu.se

*Computer Vision Laboratory, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden

†Intel Research, 3600 Juliette Lane, SC12-303, Santa Clara, CA 95054, USA

September 9, 2002

Abstract

In this paper we consider the channel representation based upon quadratic B-splines from a statistical point of view. Interpreting the channel representation as a kernel method for estimating probability density functions, we establish a channel algebra which allows to perform basic algebraic operations on measurements directly in the channel representation. Furthermore, as a central point, we identify the smoothing of channel values with a robust estimator, or equivalently, a diffusion process.

Keywords: channel representation, splines, kernel method, channel algebra, robust estimator, diffusion

Contents

1 Introduction

Since the channel representation has been introduced [13], it has proven to be a powerful framework for several kinds of applications [9, 10, 11]. Although the underlying theory is formulated more generally, the channel representation has mostly been applied in form of $\cos^2$-basis functions in the past. However, there are other possible local, smooth functions which can be used to encode a signal in channels, e.g., by B-splines. The encoding and decoding based on quadratic B-splines is introduced in section 2.

From a viewpoint of classical statistics, the channel representation can be considered as a *kernel method* for estimating the probability density function of a measurement [1]. The requirements for a channel basis function imply the necessary conditions for being a valid kernel function, in particular the monopolarity[1] constraint and the constant $\mathbb{L}_1$-norm constraint.

In practical applications it is often necessary to combine several measurements in order to obtain the desired result. As long as there is an analytic description of the required calculations, it would be quite advantageous to perform the calculations directly in the channel domain in order to maintain the approximated probability density functions. It turns out, that the spline-channels are well suited to transfer basic algebraic operations (addition, multiplication, field-multiplication, field addition[2]) to the channel representation, see section 3.

A typical application of the channel representation motivated by the interpretation as a kernel method is the *channel smoothing*. The resulting reconstructions are similar to results from diffusion processes (for an overview see [15]). Indeed, it is possible to show that *channel smoothing is related to a robust estimator*, and therefore, to a line process [2]. It is then straightforward to relate the channel smoothing to a diffusion process [3]. The relationship between channel smoothing and diffusion is worked out in section 4 as a central topic of this paper.

[1] Monopolar means positive scalar.

[2] For classical algebras there is no field-addition, because the algebra elements and the field elements are either from different sets or are not to distinguish. In our case we can add a certain constant offset to a measurement, such that we just perform a *shift* of the probability density function.

2 Channel Representation Based on B-Splines

In this section we describe how to represent a signal by means of B-spline channels.

2.1 The B-Spline Basis

B-splines are a well known technique for function approximation [6]. The basis functions of the B-splines are obtained by successive convolutions of the unit rectangle with itself [12]. The resulting basis functions become smoother and wider with each iteration (see figure 1).

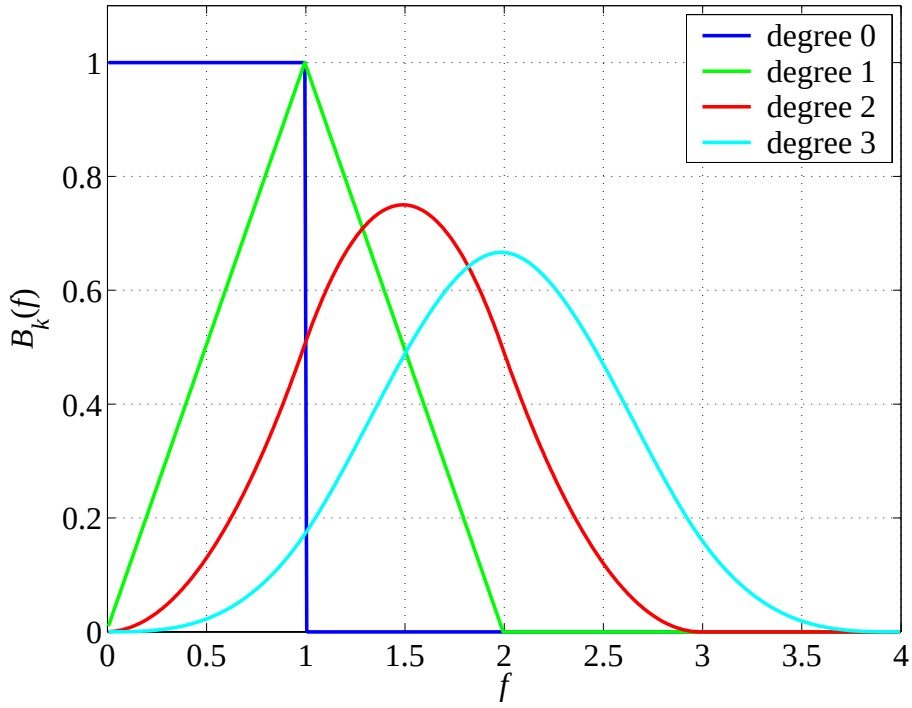

Figure 1: B-spline basis functions of degree zero to three.

The B-spline basis function (short: B-spline) of degree k is defined by convolving the unit rectangle[3]

$$\mathrm{rect}(f) = \begin{cases} 1 & \text{if } 0 \leq f \leq 1 \\ 0 & \text{else} \end{cases} \tag{1}$$

k times with itself and it is denoted by $B_k(f)$. Alternatively to the definition above, one can compute the B-splines by a recursive formula [6].

In order to obtain the full B-spline basis of degree k, the B-spline $B_k(f)$ is shifted by integers. The B-spline approximation of a function $P(f)$ is then obtained by a linear

[3]Since we apply the B-splines to signals $f(\mathbf{x})$ further below, we use f as the argument in this section. Furthermore, the spatial argument $\mathbf{x}$ is omitted if it is not relevant.

combination of the B-splines:

$$P(f) \approx \sum_{n \in \mathbb{Z}} \alpha_n B_k(f - n) \ . \tag{2}$$

Normally, the most interesting part of the spline approximation is the computation of the linear coefficients α_n. These coefficients are obtained from a large set of linear equations. In practice the sum in (2) is finite, i.e., n is an integer in $[n_{\min} - k, n_{\max} - 1]$, where $n_{\min}$ and $n_{\max}$ are the bounds of the approximation interval. Requiring

$$P(m) = \sum_{n=n_{\min}-k}^{n_{\max}-1} \alpha_n B_k(m - n) \tag{3}$$

for all $m \in \{n_{\min}, \ldots, n_{\max}\}$ yields $n_{\max} - n_{\min} + 1$ equations. However, the number of unknowns is $n_{\max} - n_{\min} + k$, and hence, we need $k - 1$ boundary conditions in order to obtain a unique solution.

2.2 Encoding with B-Splines

A signal $f(\mathbf{x})$ is encoded with the B-splines by computing $B_k(f(\mathbf{x}) - n)$. If we consider the B-splines of degree $0 - 2$, we make the following observations:

1. The B-spline basis of degree 0 corresponds to an ordinary local histogram. A single value cannot be reconstructed exactly.

2. The B-spline basis of degree 1 is the least degree basis which allows to reconstruct a single value exactly. This is done by a linear interpolation scheme.

3. The B-spline basis of degree 2 is the least degree basis which is continuously differentiable, i.e., smooth. Single values can be exactly reconstructed by a linear interpolation scheme (see below).

4. In order to have reconstructability, the B-spline basis of degree $k > 0$ requires to have an overlap of $k+1$, i.e., at almost every point $k+1$ basis functions are non-zero.

We choose the basis function for the channel representation according to the observations made above. Since we want to use a local smooth basis function, we have at least to use a basis of degree 2. Since the overlap should be small in order to reduce the computational complexity, the degree should be as small as possible. Hence, we choose $k = 2$, i.e., piecewise quadratic functions, and the overlap is given to be 3. The explicit formulation of the 2nd degree B-spline is given by

$$B_2(f) = \begin{cases} \frac{f^2}{2} & \text{if } f \in (0, 1] \\ -f^2 + 3f - \frac{3}{2} & \text{if } f \in (1, 2] \\ \frac{(3-f)^2}{2} & \text{if } f \in (2, 3) \\ 0 & \text{else} \end{cases} \ . \tag{4}$$

Hence, the value for the nth channel is obtained as

$$c_n(f) = B_2(f-n) \ . \tag{5}$$

These channel values $c_n(f)$ can be plugged into (2) in order to approximate any function $P(f)$. However, *the common point of view to the channel representation implies that* $P(f) = f$. The computation of $B_2(f-n)$ corresponds to the encoding step and the linear combination $\sum \alpha_n B_2(f-n)$ corresponds to the decoding step. Applying the decoding directly after the encoding should give the identity. For certain applications however, $P(f)$ is not necessarily given as the identity, it can also be some nonlinear mapping, e.g., logarithm, arcus-tangent, etc.. Although the decoding to a non-linear function might yield sophisticated methods for certain signal processing tasks, we only consider the linear case in this paper.

2.3 Signal Reconstruction

The question is now, which coefficients α_n yield the identity? It is easily verified, that in case of a single value the reconstruction is obtained by the following linear interpolation:

$$f = \sum_{n=n_{\min}-2}^{n_{\max}-1} (n+\frac{3}{2}) c_n(f) \ . \tag{6}$$

The linear coefficients are obtained by solving the linear system consisting of $n_{\max}-n_{\min}+1$ equations and one boundary condition. If we use $f'(n_{\min}) = 1$ as a boundary condition, we obtain the system

$$\frac{1}{2}\begin{bmatrix} -2 & 2 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 1 \end{bmatrix} \begin{pmatrix} \alpha_{n_{\min}-2} \\ \alpha_{n_{\min}-1} \\ \vdots \\ \alpha_{n_{\max}-2} \\ \alpha_{n_{\max}-1} \end{pmatrix} = \begin{pmatrix} 1 \\ n_{\min} \\ \vdots \\ n_{\max} \end{pmatrix} \ . \tag{7}$$

The first two equations can be solved separately, which gives

$$\begin{aligned} \alpha_{n_{\min}-2} &= n_{\min} - \frac{1}{2} \qquad \text{and} \\ \alpha_{n_{\min}-1} &= n_{\min} + \frac{1}{2} \ . \end{aligned}$$

For $n_{\max} > n \geq n_{\min}$ we can apply induction, assuming that $\alpha_{n-1} = n + \frac{1}{2}$. Selecting the $(n - n_{\min} + 3)$rd line in (7) gives

$$\frac{1}{2}\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{pmatrix} \alpha_{n-1} \\ \alpha_n \end{pmatrix} = \frac{\alpha_{n-1} + \alpha_n}{2} = n+1 \ .$$

Plugging in the induction assumption yields

$$\frac{n + \frac{1}{2} + \alpha_n}{2} = n+1$$

and hence

$$\alpha_n = 2(n+1) - n - \frac{1}{2} = n + \frac{3}{2} \ .$$

There are two remarks we want to make for applying (6) in practice:

1. We must take into account that in practice the channel representation is not single valued.

2. The linear interpolation assumes that the absolute scaling of the channel values remains unscaled, hence they must be normalized beforehand.

The consequence of the first remark is that we do not compute the full sum from (6), but we evaluate the sum between two different (a priori unknown) bounds, i.e., we apply a window function, the *reconstruction window*. Although we use a rectangular (Parzen) window throughout this report, the reconstruction window can have different shapes. Using the a rectangular window, the reconstruction is obtained by

$$\tilde{f} = \sum_{n=n_{\mathrm{lb}}}^{n_{\mathrm{ub}}} (n + \frac{3}{2}) c_n(f) \ , \tag{8}$$

where n_{lb} and n_{ub} are the lower bound and the upper bound of the reconstruction window, respectively.

According to the second remark, the channel values c_n must be rescaled beforehand, in order to meet the requirement

$$\sum_{n=n_{\mathrm{lb}}}^{n_{\mathrm{ub}}} c_n(f) = 1 \ . \tag{9}$$

The normalization is obtained by dividing the channel values by their sum in the reconstruction window.

This normalization of the channel values implies that any offset can be added to the reconstruction coefficients, being compensated by subtracting this offset from the final result. If w indicates the width of the reconstruction window and n_0 its lower bound, (8) can be replaced with

$$\tilde{f} = \frac{3}{2} + n_0 + \frac{w-1}{2} + \sum_{n=0}^{w-1} (n - \frac{w-1}{2}) c_{n+n_0}(f) \ . \tag{10}$$

If $w = 3$, the reconstruction is especially simple:

$$\tilde{f} = \frac{5}{2} + n_0 - c_{n_0-1}(f) + c_{n_0+1}(f) \tag{11}$$

i.e., we just apply the discrete derivative operator of size three to the channel vector.

Even if we assume that the reconstruction window is a rectangle, it is still unclear how to choose the width w and the position n_0 of the window. Ideal B-spline channel representations which correspond to single values consist of only three adjacent non-zero values. Hence a window width of three seems to be a natural choice. However, there are two cases, where we have more than three adjacent non-zero values:

1. The channel representation corresponds to multiple values and the values are sufficiently close such that the non-zero channels overlap.

2. The channel representation is non-ideal since it is modified by some operation (e.g. channel smoothing, channel addition, see below).

Whereas for the second case a wider reconstruction window is the appropriate choice, the first case requires a narrow window with an exact positioning. Before we discuss the optimal width of the reconstruction window, we turn to the positioning of the window itself.

Our decoding algorithm selects the window position with the highest (weighted) $\mathbb{L}_1$-norm[4] by by applying a (weighted) moving average filter and detecting its maximum. This method assures that single values are reconstructed correctly if the channel representation is ideal. For non-ideal representations (i.e., more than three non-zero channels), the reconstruction error is minimized, at least for a certain window size. For overlapping representations, the moving average method can either perform correctly or it leads to some averaging of neighbored values, see figure 2. The behavior depends on the distance between the two values relative to the window width.

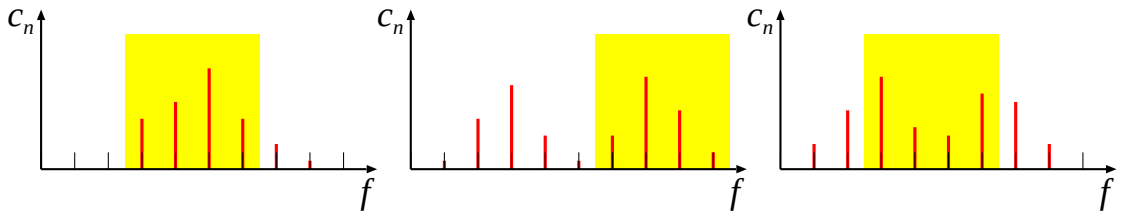

Figure 2: Reconstruction errors with a rectangular window. Left: the channel representation is too broad, the reconstructed value is too small. Center: the channels represent two values, they can be distinguished with the rectangular window. Right: the reconstruction averages the two values, they are too close.

2.4 Some Properties

In the preceding sections the encoding and decoding for the channel representation based on B-splines of degree two are defined. This section deals with some properties of the new approach and compares it to the channel representation based on $\cos^2$-functions.

In the past, the $\cos^2$-functions were chosen as basis functions since they have the nice property of yielding channel vectors with constant $\mathbb{L}_1$- *and* $\mathbb{L}_2$-norm [8]. Whereas the constant $\mathbb{L}_1$-norm is necessary to define an appropriate decoding procedure and for several applications (e.g. channel smoothing), a constant $\mathbb{L}_2$-norm is only necessary if channel vectors shall be compared by means of the scalar product [4]. It depends on the specific

[4]Notice that the channel values are monopolar, and hence, the $\mathbb{L}_1$-norm is simply the channel sum. In practice, we use a weighted norm, i.e., we apply a lowpass filter and search for the peak value.

application of the channel representation if scalar products of channel vectors occur or not.

Hence, for many applications the missing $\mathbb{L}_2$-property does not matter. On the other hand, the B-spline channel representation yields several advantages. The probably most important is the decoding by means of a linear combination, in contrast to the quite involved decoding of the $\cos^2$-channels. The linear decoding has several advantages:

1. The decoding is fast and accurate, even on machines without special numerical capabilities.

2. The channel smoothing corresponds to a linear smoothing if the signal values have small variance.

3. The decoding can directly be integrated into a neural network.

4. Analytic derivations are simplified.

5. Certain computations can be performed directly in the channel representation.

The linear decoding is obviously faster than the $\cos^2$-decoding, since linear combinations are cheaper to compute than inverse trigonometric functions.

The second advantage is illustrated by figure 3. The examples show the results from encoding three noisy 1D signals (linear function, sine, jump) with 6 channels, applying a linear smoothing to each channel, and subsequent decoding with a window of width three. Obviously, the continuous parts of all signals are quite well recovered, whereas the discontinuity in the third signal is preserved. The recovered linear function is still a linear function, i.e., the channel smoothing had the same effect as a linear smoothing in this case. We will return to this topic in section 4.

Concerning the third advantage, we just mention here that each layer of an ordinary (i.e., not higher order) neural network consists of linear combinations. These can easily realize the decoding from the channel representation.

The argument that analytic derivations are simplified by linear approaches need no further comment. The last advantage is explained in detail in the following section.

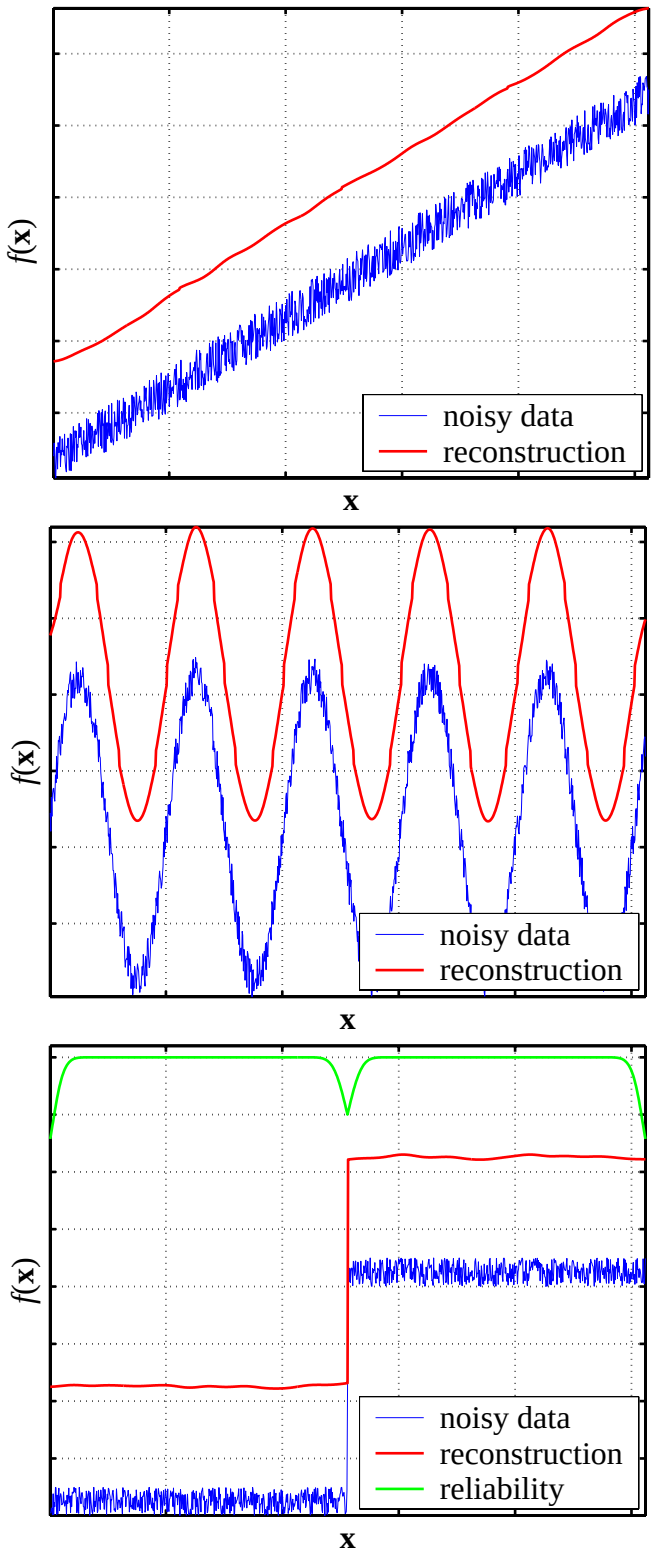

Figure 3: Three 1D experiments. Left: a linear gradient is recovered. Center: a continuous function (sine) is recovered. Right: a discontinuity is preserved. The reconstructed curves have an additional offset to improve the visualization.

3 Computing with B-Spline Channels

The aim of this section is to derive the operations on the channel representation which correspond to basic algebraic operations in the original representation. In other words, we derive a *channel algebra*. A first attempt into this direction has been made in [7], but the non-linear decoding scheme of the $\cos^2$-channels did not allow to multiply in the channel representation.

In order to describe that two channel vectors c_n and c'_m represent the same signal, i.e., the representations are isomorphic, we introduce the notation

$$c_n \simeq c'_m \qquad \text{which means} \qquad \sum_n n c_n = \sum_m m c'_m \ . \tag{12}$$

3.1 Adding Channels

The channel representation can be interpreted as an estimate of the probability density function of a certain measurement by means of a kernel-based method [1]. The B-splines of any degree obviously fulfill the requirements for being a kernel function, since

$$B_k(f) \ > \ 0 \tag{13}$$

$$\int_{\mathbb{R}} B_k(f)\, df \ = \ 1 \ . \tag{14}$$

If two independent measurements are added, their probability density functions must be convolved in order to obtain the probability density function of the sum. Applying this idea to two channel representations yields the interesting result that the reconstruction of their discrete convolution yields the sum of their reconstructions:

$$c_n(f+g) \simeq c_n(f) * c_n(g)\big|_{n \mapsto n+3/2} \ . \tag{15}$$

This result is easily shown by resorting the addends in the discrete convolution and by using the fact that the channel vector is normalized (wrt. $\mathbb{L}_1$). First, the discrete convolution reads

$$c_n(f) * c_n(g) = \sum_m c_m(f) c_{n-m}(g) \ .$$

Notice that $c_n(f) * c_n(g)$ is also normalized (wrt. $\mathbb{L}_1$). Applying (6) to $c_n(f) * c_n(g)$ yields

$$\begin{aligned}
\frac{3}{2} + \sum_n n(c_n(f) * c_n(g)) \ &= \ \frac{3}{2} + \sum_n n \sum_m c_m(f) c_{n-m}(g) \\
\text{substitute } l = n - m \qquad &= \ \frac{3}{2} + \sum_l \sum_m (l+m) c_m(f) c_l(g) \\
&= \ \frac{3}{2} + \sum_l l c_l(g) \sum_m c_m(f) + \sum_l c_l(g) \sum_m m c_m(f) \\
&= \ \frac{3}{2} + (f - \frac{3}{2}) \cdot 1 + 1 \cdot (g - \frac{3}{2}) \\
&= \ f + g - \frac{3}{2} \ .
\end{aligned}$$

If we change the index n to $n+\frac{3}{2}$, $c_n(f) * c_n(g)$ is decoded to the same signal as $c_n(f+g)$.

In this derivation channel indices are shifted. Actually, we can use index-shift and index-scaling to simulate 1D affine transforms to the encoded signal.

3.2 Affine Transforms of Channels

Affine transforms correspond to field addition and field multiplication. Field multiplication means a scaling of the channel positions and likewise a scaling of the approximated probability density function. Field addition means a shift of the channel positions.

First, we consider offsets, i.e., field addition. Adding a constant $r \in \mathbb{R}$ to the channel index gives:

$$c'_m(f) = c_n(f)\big|_{n\mapsto n+r} \simeq c_n(f+r) \ . \tag{16}$$

Inserting $c'_m(f)$ in (6) yields

$$\sum_m (m+\frac{3}{2}) c'_m(f) = \sum_n (n+r+\frac{3}{2}) c_n(f) = r + \sum_n (n+\frac{3}{2}) c_n(f) = r + f \ .$$

More interesting is the scaling of the channel index (field multiplication), since this allows to alter the dynamics of the decoded signal. Multiplying the index by a positive constant $r \in \mathbb{R}^+$ results in

$$c'_m(f) = c_n(f)\big|_{n\mapsto rn} \simeq c_n(rf + \frac{3(1-r)}{2}) \ . \tag{17}$$

Inserting $c'_m(f)$ in (6) yields

$$\begin{aligned}\sum_m (m+\frac{3}{2}) c'_m(f) &= \frac{3(1-r)}{2} + \sum_n (rn + r\frac{3}{2}) c_n(f) \\ &= \frac{3(1-r)}{2} + r\sum_n (n+\frac{3}{2}) c_n(f) = \frac{3(1-r)}{2} + rf \ .\end{aligned}$$

The resulting implicit shift by $\frac{3(1-r)}{2}$ can easily be compensated by another shift operation.

Finally, a signal can be reflected by negating the index:

$$c'_m(f) = c_n(f)\big|_{n\mapsto -n} \simeq c_n(-f+3) \ . \tag{18}$$

Inserting $c'_m(f)$ in (6) yields

$$\sum_m (m+\frac{3}{2}) c'_m(f) = \sum_n (-n+\frac{3}{2}) c_n(f) = 3 - \sum_n (n+\frac{3}{2}) c_n(f) = 3 - f \ .$$

3.3 Multiplying Channels

In order to multiply two measurements by means of their channel representations, it is necessary to introduce some new notations. Let $c_n(f)$ be the channel values of the representation of a measurement f. We define

$$c_{n/m} = \frac{c_{\lfloor n/m \rfloor}(\lceil n/m \rceil - n/m) + c_{\lceil n/m \rceil}(n/m - \lfloor n/m \rfloor)}{m} \quad , \tag{19}$$

where $\lfloor \cdot \rfloor$ is the next smaller integer (floor operation), $\lceil \cdot \rceil$ is the next larger integer (ceiling operation), and m is a positive integer.

The preceding definition is reasonable since the measurement f is recovered by

$$f = \frac{3}{2} + \sum_n \frac{n}{m} c_{n/m}(f) \quad . \tag{20}$$

By some straightforward calculations, we obtain

$$\begin{aligned}
\frac{3}{2} + \sum_n \frac{n}{m} c_{n/m}(f) &= \frac{3}{2} + \frac{1}{m^2} \sum_n n(c_{\lfloor n/m \rfloor}(\lceil n/m \rceil - n/m) + c_{\lceil n/m \rceil}(n/m - \lfloor n/m \rfloor)) \\
&\quad \text{substitue } n = mk + l \\
&= \frac{3}{2} + \frac{1}{m^2} \sum_k \sum_{l=0}^{m-1} (mk + l)(c_k(1 - l/m) + c_{k+1} l/m) \\
&= \frac{3}{2} + \frac{1}{m^2} \left(\sum_{l=0}^{m-1} \left((l - l^2/m) \sum_k c_k + l^2/m \sum_k c_{k+1} \right. \right. \\
&\qquad \left. \left. + (m - l) \sum_k k c_k + l \sum_k k c_{k+1} \right) \right) \\
&= \frac{3}{2} + \frac{1}{m^2} \left(\sum_{l=0}^{m-1} \left(l + m \sum_k k c_k - l \sum_k c_{k+1} \right) \right) = \frac{3}{2} + \sum_k k c_k \quad .
\end{aligned}$$

Using the notation (19), we define the following, convolution-like operation:

$$c_n \odot c'_n = \sum_m c_m c'_{n/m} \quad . \tag{21}$$

Using this definition, we can show the following central equation:

$$c_n(fg + \frac{3}{2}) \simeq c_n(f)\big|_{n \mapsto n+3/2} \odot c_n(g)\big|_{n \mapsto n+3/2} \quad . \tag{22}$$

Decoding the right side of (22) yields

$$\begin{aligned}
& \frac{3}{2} + \sum_n n \sum_m c_{m-3/2}(f) c_{n/m-3/2}(g) \\
= \; & \frac{3}{2} + \sum_n \sum_m (m + \frac{3}{2}) c_m(f) \frac{n}{m} c_{n/m-3/2}(g) \\
= \; & \frac{3}{2} + \sum_m \sum_n \left(m + \frac{3}{2} \right) c_m(f) \left(\frac{n}{m} + \frac{3}{2} \right) c_{n/m}(g) \\
= \; & \frac{3}{2} + \sum_m \left(\frac{3}{2} c_m(f) + m c_m(f) \right) \left(\frac{3}{2} + \sum_n \frac{n}{m} c_{n/m}(g) \right) \\
= \; & \frac{3}{2} + \sum_m \left(\frac{3}{2} c_m(f) + m c_m(f) \right) g = \frac{3}{2} + fg \;\; .
\end{aligned}$$

Hence, the product of two measurements can be computed by means of (22) with a subsequent shift by $-\frac{3}{2}$. This concludes the section about the channel algebra. Applying the derived formulas, it is basically possible to compute all kinds of linear combinations of arbitrary order products of measurements. In any of these operations, the probability density function is altered in the correct way.

4 Channel Smoothing and Diffusion

A straightforward application of the channel representation is image enhancement by means of *channel smoothing*. In order to denoise an image, it is encoded in channels and each channel is smoothed by a linear filter. Depending on the applied filter, the decoded signal is a denoised or even simplified version of the original image, see figure 4. In these examples the image was encoded in 24 channels and each channel was smoothed with a binomial filter of width 9. The reconstruction is calculated with a window size of three.

In the following sections, we relate the channel smoothing to the approach of non-linear diffusion.

4.1 Channel Smoothing

Smoothing the channels with a linear filter is equivalent to smoothing the original signal if, and only if, the signal is smooth to a certain degree (see section 2.4). The smoothness constraint can be stated more precisely: the signal is smooth with respect to a certain filter and to its channel representation, if all channel values in the filter support, which are neglected in the reconstruction, are zero. Equivalently, the signal is smooth if no non-zero values are neglected in the reconstruction of the filtered channels. These definitions of smoothness are illustrated in figure 5.

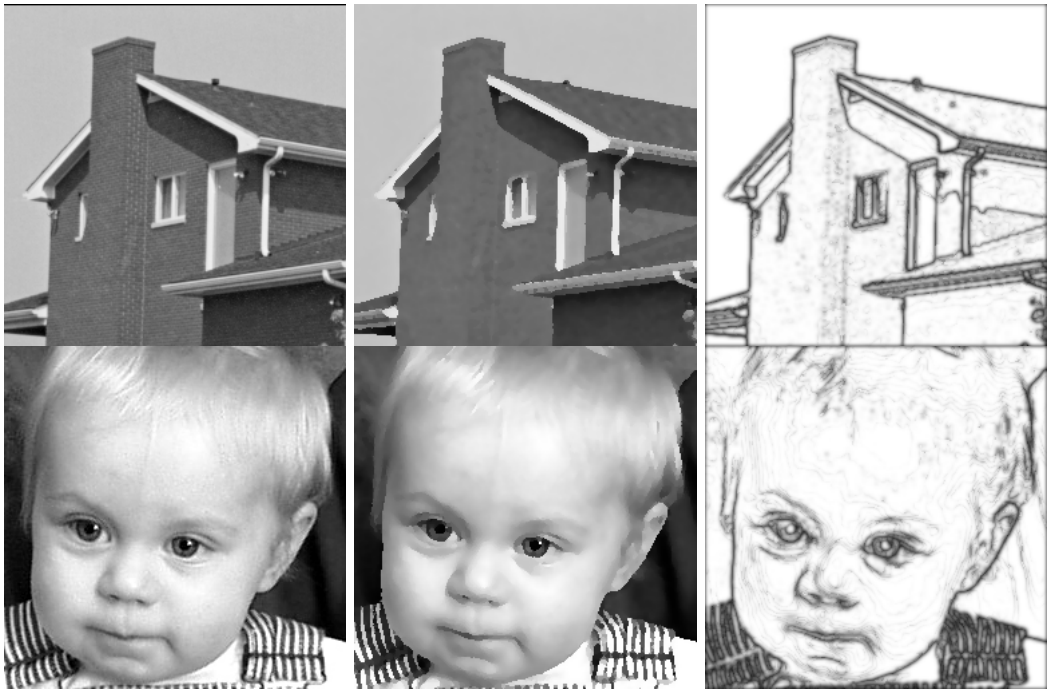

Figure 4: 2D experiments. Left column: original images. Center column: reconstruction. Right column: reliability of the reconstruction.

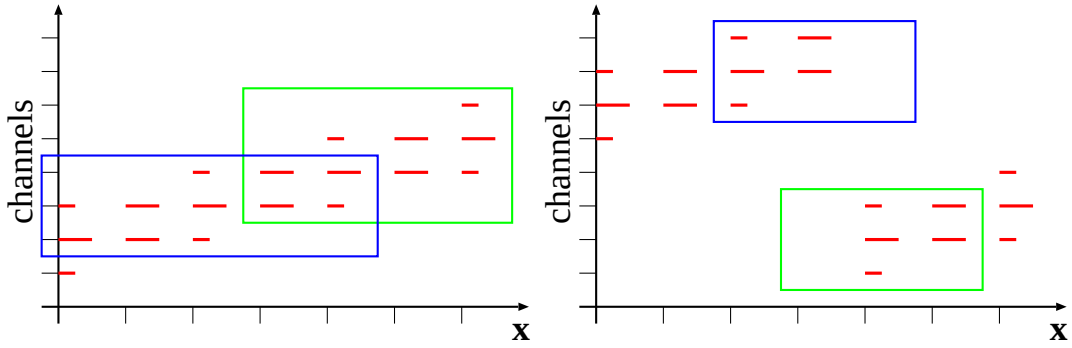

Figure 5: Smoothness of channel representations. Left: a continuous gradient. The blue frame indicates a smoothing filter of size 5 and a reconstruction window of size 3. Obviously, there are two non-zero channel values in the filter support which do not lie inside the frame, i.e., they do not effect the reconstruction. The signal is not smooth with respect to the parameters indicated by the blue frame. According to the parameters indicated by the green frame (filter size 4 and reconstruction window size 4), the signal is smooth. Right: at the discontinuity the frame jumps from the blue position to the green position. Unless the reconstruction window covers the whole dynamic of the discontinuity, discontinuous signals are not smooth.

If a signal is smooth, filtering the channels and subsequent reconstruction is equivalent to directly filtering the signal. This result is easily obtained by linearity.

If the signal is smooth, the reconstruction formulae (6) and (8) are equivalent since no non-zero channel values are neglected in (8). Hence, the reconstruction of the smoothed channels reads

$$\sum_{n=n_{\min}-2}^{n_{\max}-1} (n+\frac{3}{2}) h(\mathbf{x}) * c_n(f(\mathbf{x})) = h(\mathbf{x}) * \sum_{n=n_{\min}-2}^{n_{\max}-1} (n+\frac{3}{2}) c_n(f(\mathbf{x})) = h(\mathbf{x}) * f(\mathbf{x}) \ ,$$

where $h(\mathbf{x})$ is a smoothing kernel.

If the signal violates the smoothness condition, the reconstruction from the smoothed channels is no longer identical to the smoothed signal. However, if the deviation from the smoothness constraint is small, the difference between the reconstruction and the smoothed signal is small as well. This difference is increasing with the deviation from the smoothness condition until the signal contains a discontinuity. In this case, the channel smoothing results in a totally different result than the smoothing of the signal. The channel smoothing does not smooth over discontinuities, i.e., instationary signal parts. This is a correct behavior since it is only admissible (from a stochastic point of view) to smooth stationary signals.

The just described behavior will be formalized below, the informal description is illustrated in figure 6. The original signal in this experiment is a 1D gradient with variable

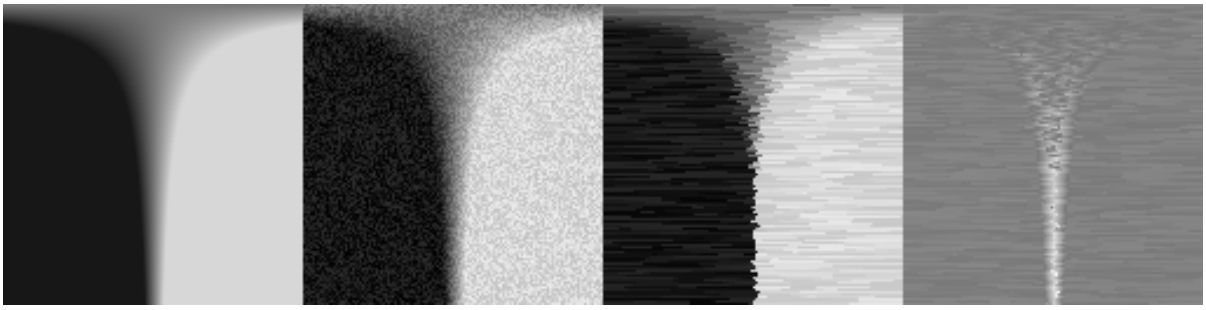

Figure 6: Experiment to verify the behavior for reduced smoothness. From left to right: original 1D signal family, noisy 1D signal family, results from channel smoothing, absolute differences.

steepness. We varied the slope in a certain interval, obtaining a signal family. Aligning the signals vertically above each other yields the image in the left of figure 6. Noise is added to the image and the signal is encoded to channels. Each channel is smoothed line-wise (i.e., in the direction of the original signal). Afterwards the channels are decoded (second image from the right). The absolute difference between the reconstruction and the original signal shows that the smoothness of the reconstruction decreases with increasing gradient. Hence, the smoothing within the region of the gradient also decreases.

4.2 Robust Estimation

The behavior which is described in the previous section can be explained by means of robust statistics. Assume that in a certain region the mean value of the signal is given. If we add another data sample, the mean value will change according to a certain *influence function* [2]. For ordinary $\mathbb{L}_2$-optimization, the influence function is simply the difference between sample and mean. In robust statistics however, we have a different behavior. If the deviation from the mean is small, there is no difference to $\mathbb{L}_2$-optimized estimation. For large deviations however, the influence of the new sample is suppressed; it is considered to be an *outlier*.

Smoothing with a Gaussian kernel directly corresponds to the solution of a least-squares problem, since a linear influence function implies the linear, isotropic heat equation [2]. In case of channel smoothing with a Gaussian kernel, each channel is a solution of the least-squares problem, but how about the reconstructed signal?

In order to understand the process acting on the whole signal, we consider the reconstruction (10) more in detail. Basically, the reconstruction is a discrete convolution of the B-splines with a finite symmetric ramp function $R(n)$:

$$\tilde{f} = K + \sum_n R(n) B_2(f - n) \ .$$

The resulting *continuous* function $\tilde{f}(f)$ is plotted for different window widths in figure 7.

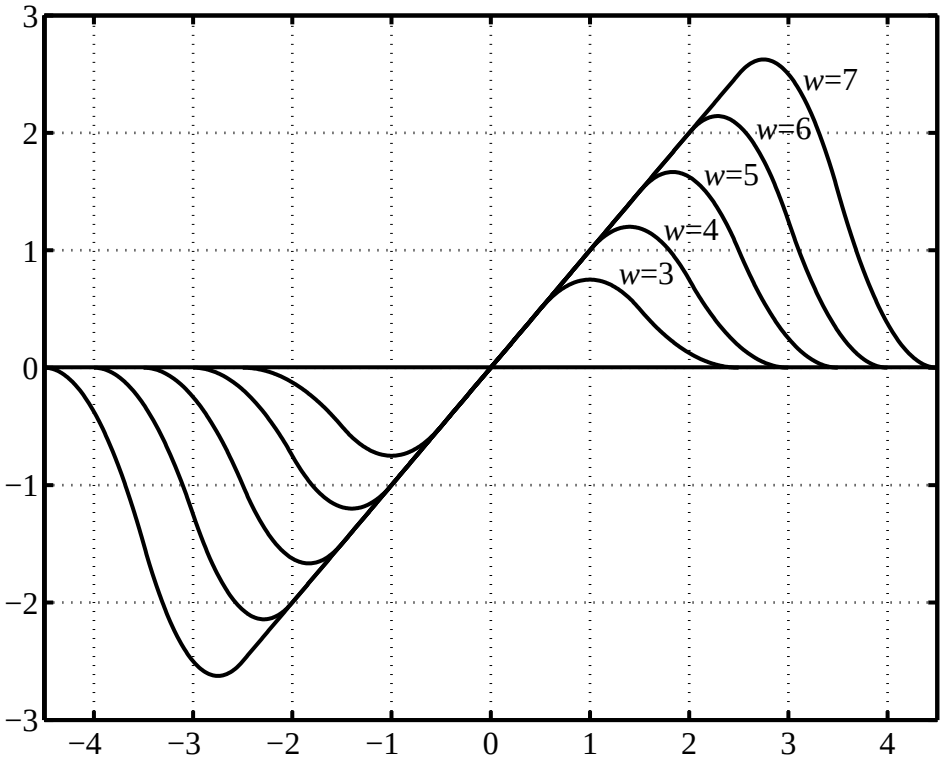

Figure 7: Influence functions for different reconstruction window widths (3-7). The reconstruction window is fixed symmetric to the origin.

This function is just the influence function mentioned above. Close to the origin, this influence function is linear like in the least-squares case. For large $|f|$ it is zero, i.e., f is considered as an outlier and it is neglected. The regions in between form continuous transitions. Hence, *channel smoothing can be considered as a solution of a robust estimation problem.*

The influence function of channel smoothing can be computed analytically. However, the formulation depends on the window width and a general description is quite involved. Thus, we calculate one special case for the window width three. The reconstruction formula is given by (11). With an appropriate shift of the origin, we obtain the following result:

$$\psi(f) = \tilde{f}(f) = \begin{cases} -B_2(f+\frac{5}{2}) = -\frac{f^2}{2} - \frac{5f}{2} - \frac{25}{8} & \text{if } f \in (-2.5, -1.5) \\ -B_2(f+\frac{5}{2}) = f^2 + 2f + \frac{1}{4} & \text{if } f \in [-1.5, -0.5) \\ -B_2(f+\frac{5}{2}) + B_2(f+\frac{1}{2}) = f & \text{if } f \in [-0.5, -0.5] \\ B_2(f+\frac{1}{2}) = -f^2 + 2f - \frac{1}{4} & \text{if } f \in (0.5, 1.5] \\ B_2(f+\frac{1}{2}) = \frac{f^2}{2} - \frac{5f}{2} + \frac{25}{8} & \text{if } f \in (1.5, 2.5) \\ 0 & \text{else} \end{cases} . \tag{23}$$

This function with its respective parts is plotted in figure 8.

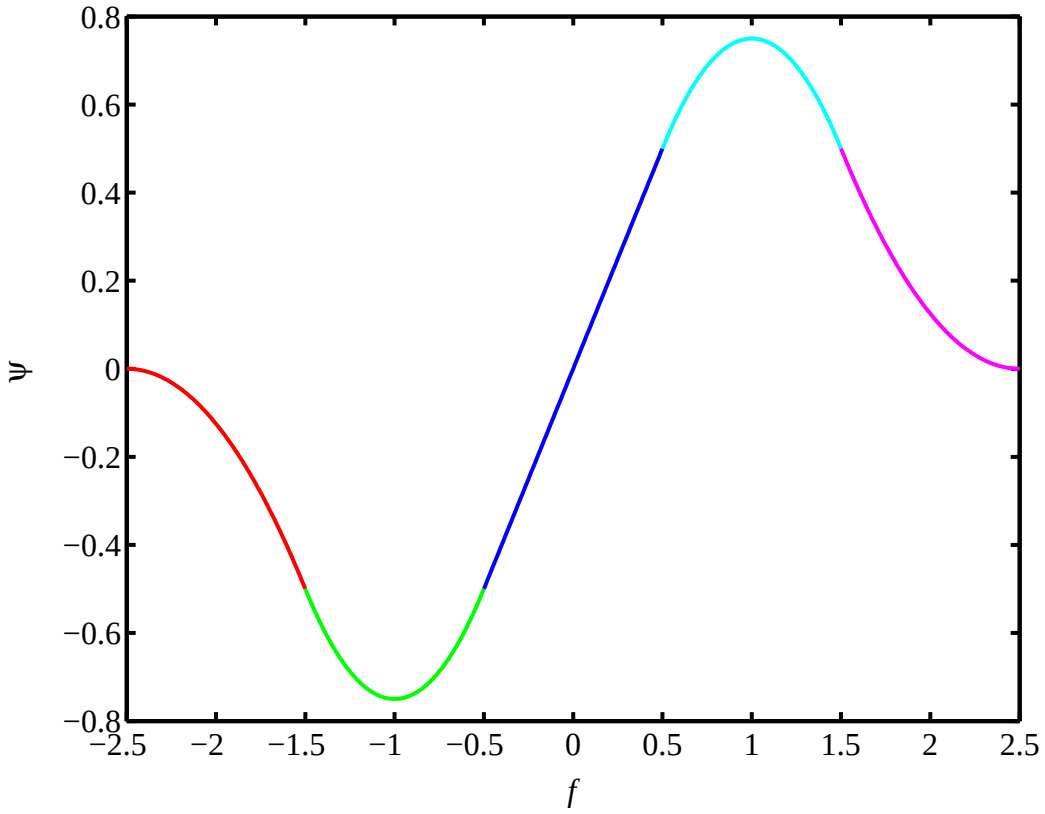

Figure 8: The influence function for reconstruction window width three consists of five parts (see (23)).

Integrating the influence function yields the error norm of the robust estimator [2], where we have to add an offset such that the error norm for $f = 0$ is zero, i.e.,

$$\rho(f) = \int_{-\infty}^{f} \psi(f')\,df' - \int_{-\infty}^{0} \psi(f')\,df' \quad . \tag{24}$$

Plugging in (23) yields

$$\rho(f) = \begin{cases} -\frac{f^3}{6} - \frac{5f^2}{4} - \frac{25f}{8} - \frac{79}{48} & \text{if } f \in (-2.5, -1.5) \\ \frac{f^3}{3} + f^2 + \frac{f}{4} + \frac{1}{24} & \text{if } f \in [-1.5, -0.5) \\ \frac{f^2}{2} & \text{if } f \in [-0.5, 0.5] \\ -\frac{f^3}{3} + f^2 - \frac{f}{4} + \frac{1}{24} & \text{if } f \in (0.5, 1.5] \\ \frac{f^3}{6} - \frac{5f^2}{4} + \frac{25f}{8} - \frac{79}{48} & \text{if } f \in (1.5, 2.5) \\ \frac{23}{24} & \text{else} \end{cases} \quad . \tag{25}$$

The error norm is plotted in figure 9. The channel smoothing corresponds to a minimization of (25).

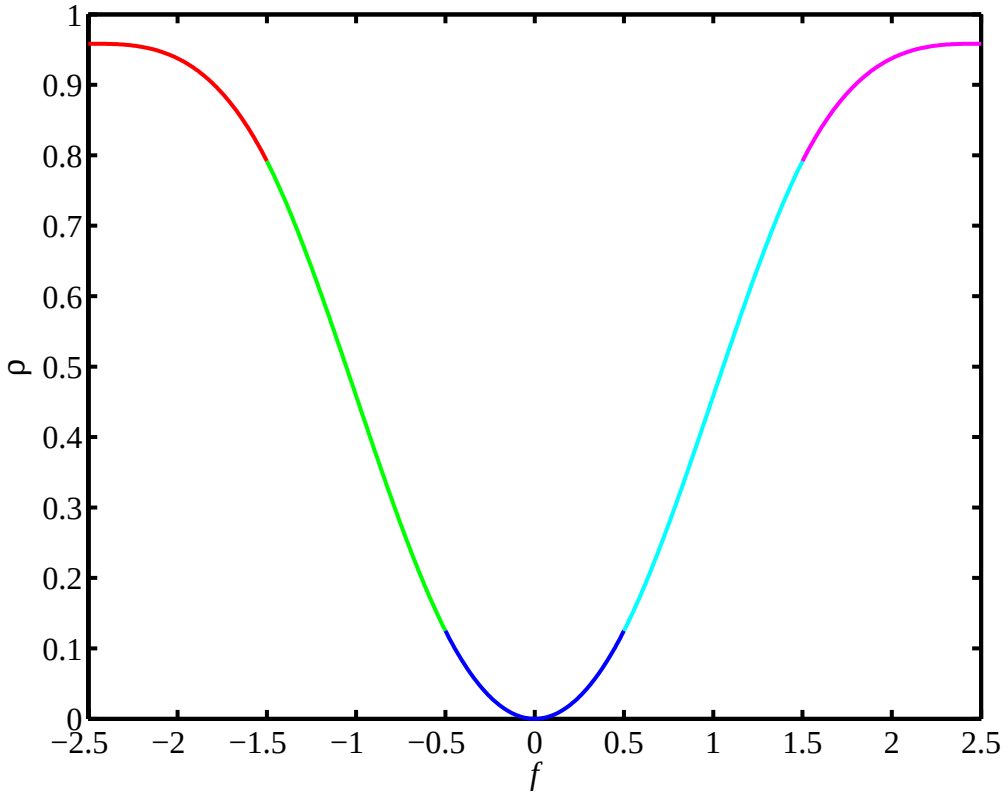

Figure 9: Error norm for window width three.

4.3 Outlier Processes

In [2] the authors present a method for converting a robust estimator into the objective function of an outlier process. Applying this method to our robust estimator obtained from the channel smoothing yields the following results.

In order to formulate an outlier process z, it is necessary to calculate the discontinuity penalty function $\Psi(z)$. The first step is to rewrite the error norm by plugging in $x = \sqrt{2\omega}$ for $\omega > 0$:

$$\phi(\omega) = \rho(\sqrt{2\omega}) = \begin{cases} \omega & \text{if } \omega \in [0, \frac{1}{8}] \\ -\frac{2\sqrt{2}\omega^{3/2}}{3} + 2\omega - \frac{\sqrt{2}\omega^{1/2}}{4} + \frac{1}{24} & \text{if } \omega \in (\frac{1}{8}, \frac{9}{8}] \\ \frac{\sqrt{2}\omega^{3/2}}{3} - \frac{5\omega}{2} + \frac{25\sqrt{2}\omega^{1/2}}{8} - \frac{79}{48} & \text{if } \omega \in (\frac{9}{8}, \frac{25}{8}) \\ \frac{23}{24} & \text{if } \omega \geq \frac{25}{8} \end{cases} .$$

In a second step, it is necessary to compute the first and second partial derivatives of $\phi(\omega)$ wrt. ω:

$$\phi'(\omega) = \begin{cases} 1 & \text{if } \omega \in [0, \frac{1}{8}] \\ -\sqrt{2}\omega^{1/2} + 2 - \frac{\sqrt{2}\omega^{-1/2}}{8} & \text{if } \omega \in (\frac{1}{8}, \frac{9}{8}] \\ \frac{\sqrt{2}\omega^{1/2}}{2} - \frac{5}{2} + \frac{25\sqrt{2}\omega^{-1/2}}{16} & \text{if } \omega \in (\frac{9}{8}, \frac{25}{8}) \\ 0 & \text{if } \omega \geq \frac{25}{8} \end{cases} ,$$

$$\phi''(\omega) = \begin{cases} 0 & \text{if } \omega \in [0, \frac{1}{8}] \\ -\frac{\sqrt{2}}{16}\frac{8\omega-1}{\omega^{3/2}} & \text{if } \omega \in (\frac{1}{8}, \frac{9}{8}] \\ \frac{\sqrt{2}}{32}\frac{8\omega-25}{\omega^{3/2}} & \text{if } \omega \in (\frac{9}{8}, \frac{25}{8}) \\ 0 & \text{if } \omega \geq \frac{25}{8} \end{cases} .$$

In the third step, we have to check several requirements. In particular, we have $\phi'(0) = 1$, $\lim_{\omega\to\infty} \phi'(\omega) = 0$, and $\phi''(\omega) < 0$ for $\omega \in (\frac{1}{8}, \frac{25}{8})$.[5] Defining $z = \phi'(\omega)$ (fourth step), we compute the inverse of $\phi'(\omega)$ (step five):

$$(\phi')^{-1}(z) = \begin{cases} z^2 + (5 - (z(5+z))^{1/2})z + \frac{25}{8} - \frac{5}{2}(z(5+z))^{1/2} & \text{if } z \in [0, \frac{1}{3}] \\ \frac{7}{8} - z + \frac{1}{2}(3 - 4z + z^2)^{1/2} + \frac{z^2}{4} - \frac{z}{4}(3 - 4z + z^2)^{1/2} & \text{if } z \in (\frac{1}{3}, 1] \end{cases} .$$

In step six, the penalty function $\Psi(z)$ is defined by

$$\Psi(z) = \phi((\phi')^{-1}(z)) - z(\phi')^{-1}(z) \ , \tag{26}$$

which gives an analytic expression which is too long to be presented in a reasonable way. The graph of this expression is plotted in figure 10, together with the plot of the (scaled) penalty function of Tukey's biweight (see [2]). Obviously, the penalty functions (and therefore the robust estimators) do not differ essentially.

The seventh and last step in the method of [2] is to formulate the objective function of the line process, which reads

$$E(f, z) = \frac{f^2 z}{2} + \Psi(z) \ . \tag{27}$$

[5] In [2] the requirement reads $\phi''(\omega) < 0$, i.e., for the whole interval, but in their examples the authors find $\phi''(\omega) < 0$ on a smaller interval being sufficient.

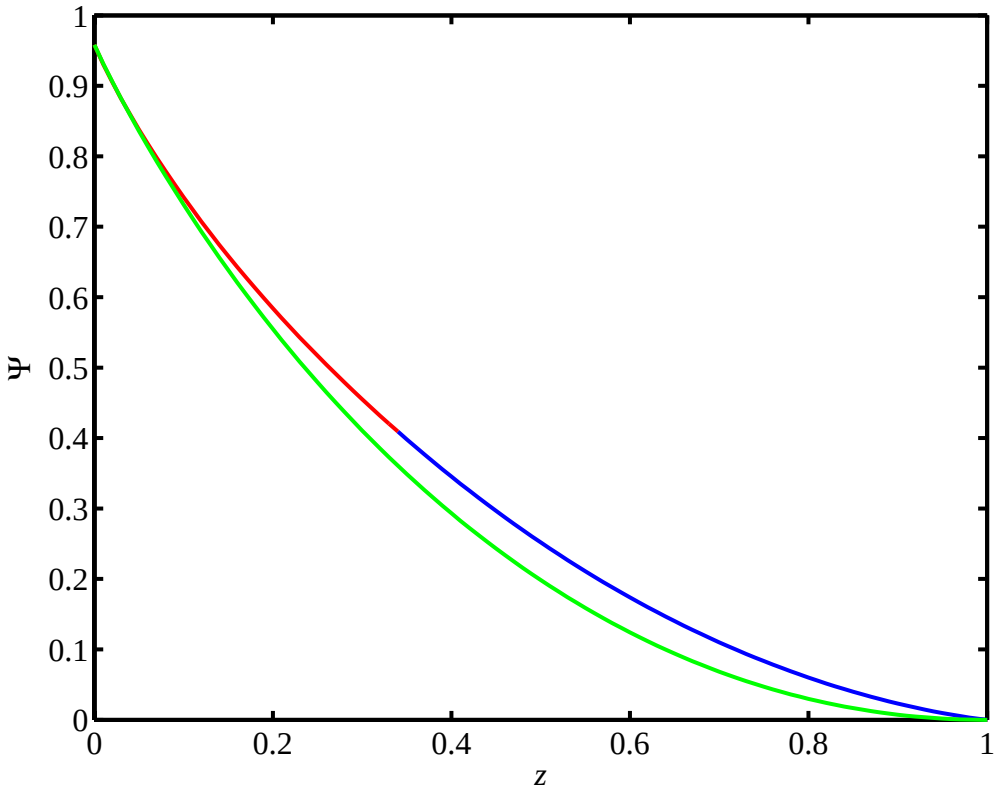

Figure 10: Penalty function of the channel smoothing (red/blue) compared to the penalty function of Tukey's biweight (green).

The objective function is illustrated in figure 11. If we minimize $E(f, z)$ wrt. z, we should obtain the original error norm again. The infimum of $E(f, z)$ is plotted in figure 12, which is identical to $\rho(f)$ as required [2].

4.4 Relation to Diffusion

The relation between line process formulation, robust estimator, and diffusion is explained in e.g. [3]. Nonlinear diffusion can either be considered as a minimization of a line process, or by means of a robust estimator. In the latter case, we obtain the diffusion equation according to

$$\frac{\partial f(\mathbf{x}, t)}{\partial t} = \operatorname{div}\left(\rho'(|\nabla f|)\frac{\nabla f}{|\nabla f|}\right) \quad . \tag{28}$$

According to the derivation in the previous section, the channel smoothing basically corresponds to a diffusion applying Tukey's biweight influence function. In [2] it is claimed that the exact formulation of the influence function is not critical as long as the penalty function has a similar shape. Hence, the influence function (23) is as justified as any other known influence function with a similar shape (e.g. Tukey, MFT, Leclerc, Geman & McClure). Furthermore, diffusion based on Tukey's function performs better than e.g. Perona-Malik diffusion [3]. Hence, the channel smoothing is supposed to produce quite

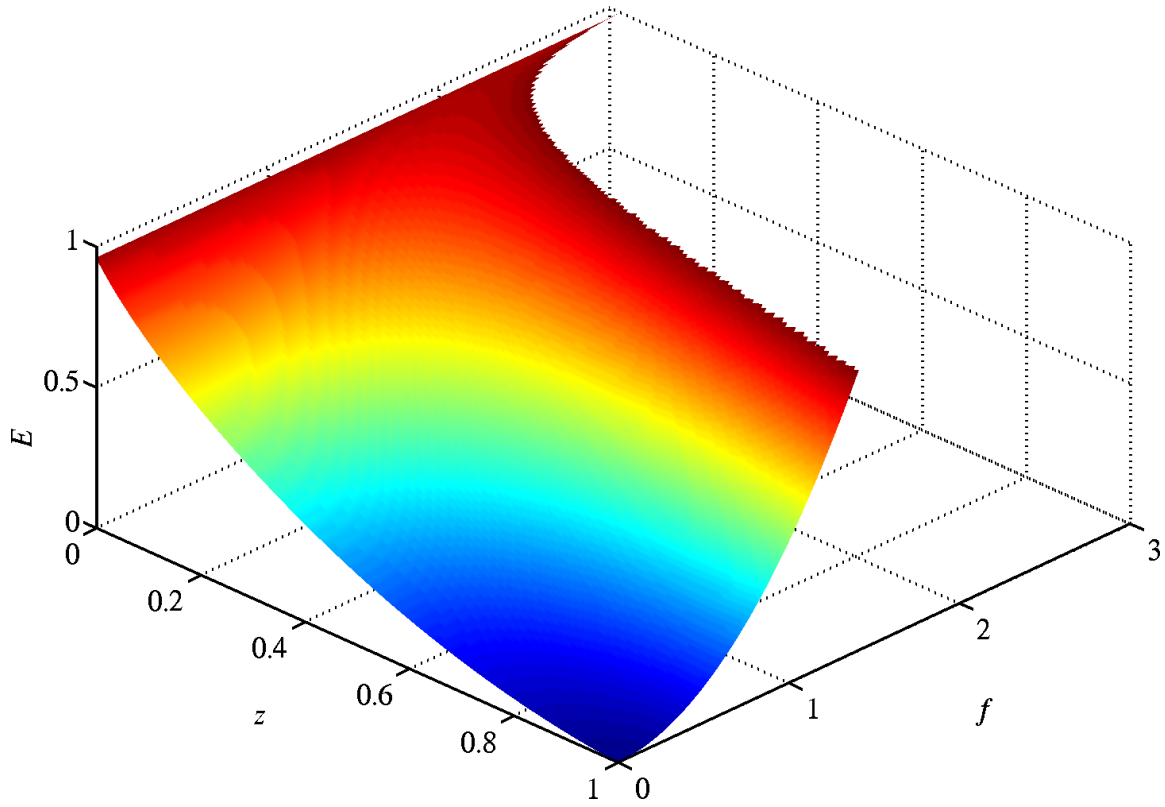

Figure 11: Objective function of the channel smoothing as a 2D plot.

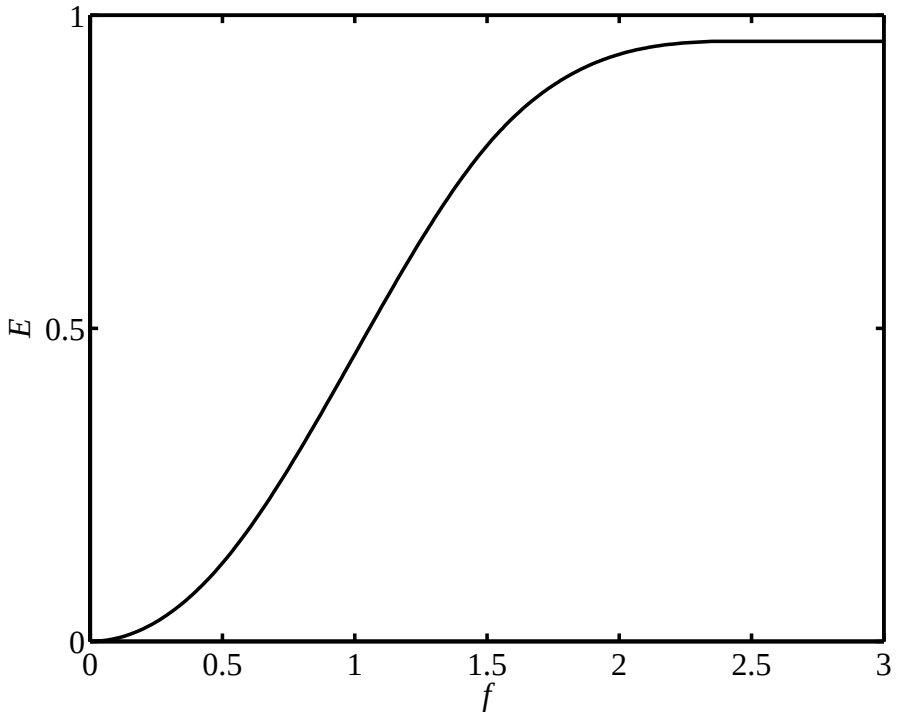

Figure 12: Infimum for $0 \leq z \leq 1$ of the objective function. Compare to figure 9.

good results. Figure 13 shows the results of an isotropic but non-linear method [5] and of the channel smoothing. The results can be directly compared to the one of an anisotropic method [14] (see [16]).

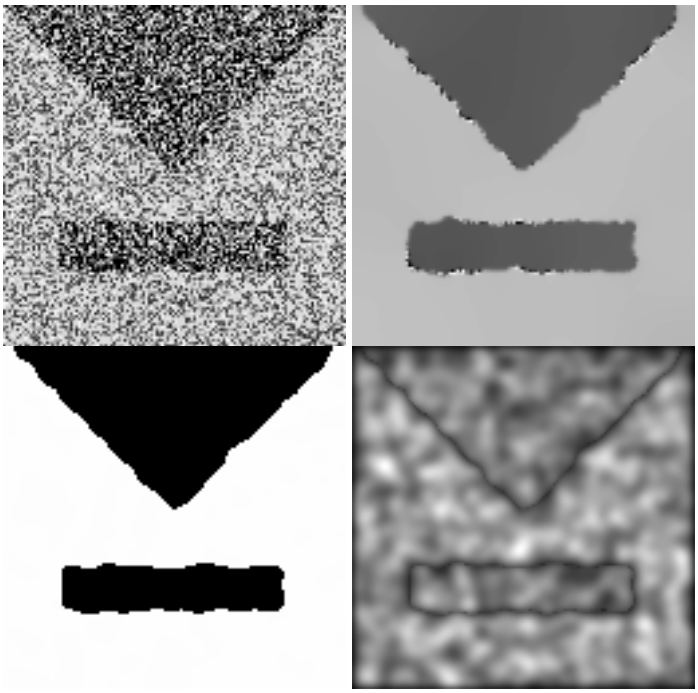

Figure 13: Experiment according to [16]. Upper row left: original noisy signal; right: enhanced signal using isotropic, non-linear diffusion [5]. Bottom row left: enhanced signal from channel smoothing; right: reliability of the reconstruction.

The result from the isotropic method is obtained with the following parameters: contrast 5, noise scale 2, time step 0.25, and 500 iterations. The result from channel smoothing is obtained with the following parameters: 50 channels, binomial smoothing filter (size 25), decoding with a window width of three.

Comparing channel smoothing and non-linear diffusion yields a couple of pros and cons. We just want to mention some of them at this place:

- The channel smoothing is faster, at least for long diffusion times.
- The channel smoothing is easier to implement for parallel computing.
- The channel smoothing does not show an effect of discretized diffusion time.
- But: the channel smoothing shows some quantization effect.

Whereas the first three points are more or less obvious, the want to discuss the last point, which is actually the only drawback of channel smoothing that we have found up to now. This 'quantization effect' only occurs in certain situation. First, the channels must represent a multiple-valued signal, i.e., the effect only occurs at discontinuities. Second,

the difference between the signal values must lie in a certain (small) interval, such that it is neither so small that the values should be linearly interpolated, nor so big that one of the values is suppressed. Hence, we are in the non-linear, non-zero interval of the influence function.

Since the influence function is placed at discrete points (channel positions), the actual influence of an outlier depends on the positioning of the channels. This effect is visualized in figure 14. In the experiment, two different linear functions are encoded in ten channels.

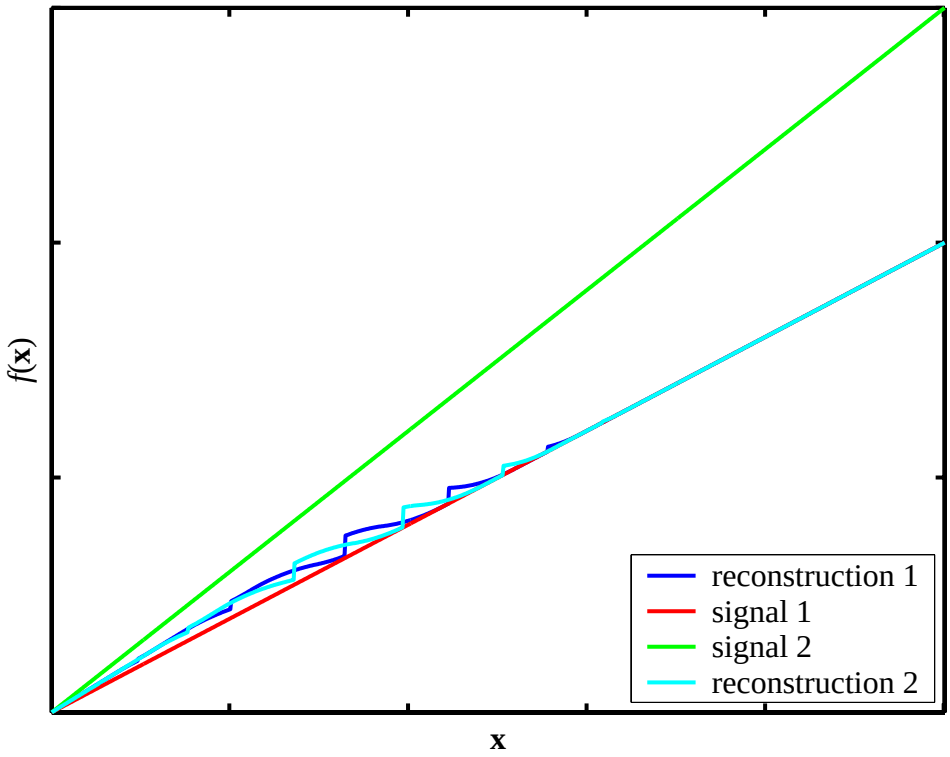

Figure 14: Quantization effect of the channel representation. The channels contain the channel values of signal 1 and signal 2. The decoding yields a linear interpolation for small differences and a selection of signal 1 for large distances. In between, the exact behavior depends on the channel positions as it can be seen by the two different reconstructions.

The two representations are added component-wise, where the coefficients of the first signal are weighted with two. The subsequent reconstruction yields a linear average for small distances of the two functions. For large distances, the first signal is recovered. In between, there should be a continuous transition between the two cases, but some distortions occur. These distortions depend on the channel positions, which can be seen in the two reconstructions based on two representations that only differ with respect to their channel positions.

Although these distortions occur, they can mostly be neglected. First, they are bounded to be less than one half the channel width, i.e., for a gray-scale image with 256 gray values and 32 channels, the maximal error is 4 gray values. Second, if the channels of the two (or more) signals have different weights, the distortion is further re-

duced. This is typically the case for e.g. image enhancement (removing noise). Third, the distortion can only occur at places where multiple values exist which have a certain critical distance. This distance depends on the reconstruction window width. Finally, the distortion can be reduced by increasing the overlap between the basis functions, which corresponds to oversampling the channels [8].

5 Future Work

In some future work we are going to exploit the introduced methods from the channel algebra. According to usability, the algorithm for addition and multiplication has to be designed such that the number of channels does not increase too much. In this context, it would be advantageous to have some kind of channel compression technique if the estimated probability functions become too broad. The compression of channels can basically be achieved by subsampling, where the correct reconstruction must be assured.

The channel smoothing will also be applied in this context, yielding robust estimates of certain measurements. According to the quantization effect, we will make further investigations in order to reduce the error. Furthermore, we will extend the theoretic considerations to higher dimensional feature spaces with certain topologies, e.g., orientation vectors, color vectors, etc.. Quite unclear at this point is the relation of the channel smoothing to anisotropic methods. Basically, it shows an anisotropic behavior. At edges, the channel representation blurs parallel to the edge but not perpendicular to the edge.

Acknowledgments

We appreciate the help of Joachim Weickert for providing the code for the isotropic, non-linear diffusion and the test image in figure 13.
This work has been supported by DFG Grant FE 583/1-1 (M.Felsberg).

References

[1] Bishop, C. M. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[2] Black, M. J., and Rangarajan, A. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *International Journal of Computer Vision 19*, 1 (1996), 57–91.

[3] Black, M. J., Sapiro, G., Marimont, D. H., and Heeger, D. Robust anisotropic diffusion. *IEEE Transactions on Image Processing 7*, 3 (1998), 421–432.

[4] Borga, M. *Learning Multidimensional Signal Processing*. PhD thesis, Linköping University, Sweden, 1998.

[5] Catté, F., Lions, P.-L., Morel, J.-M., and Coll, T. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J. Numer. Analysis 32* (1992), 1895–1909.

[6] de Boor, C. *Splinefunktionen*. Birkhäuser, 1990.

[7] Felsberg, M. Disparity from monogenic phase. In *24. DAGM Symposium Mustererkennung, Zürich* (2002), L. v. Gool, Ed., Lecture Notes in Computer Science, Springer-Verlag, Heidelberg. to appear.

[8] Forssén, P.-E. Sparse representations for medium level vision. Lic. Thesis LiU-Tek-Lic-2001:, Dept. EE, Linköping University, February 2001.

[9] Forssén, P.-E., and Granlund, G. H. Sparse feature maps in a scale hierarchy. In *Proc. Int. Workshop on Algebraic Frames for the Perception-Action Cycle* (Kiel, Germany, September 2000), G. Sommer and Y. Y. Zeevi, Eds., vol. 1888 of *Lecture Notes in Computer Science*, Springer-Verlag, Heidelberg.

[10] Forssén, P.-E., Granlund, G. H., and Wiklund, J. Channel representation of colour images. Tech. Rep. LiTH-ISY-R-2418, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, 2002.

[11] Granlund, G. H., Forssén, P.-E., and Johansson, B. HiperLearn: A High Performance Learning Architecture. Tech. Rep. LiTH-ISY-R-2409, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, January 2002. Submitted to IEEE Trans. on Neural Networks.

[12] Jähne, B. *Digitale Bildverarbeitung*. Springer-Verlag, Berlin, 1997.

[13] Nordberg, K., Granlund, G. H., and Knutsson, H. Representation and Learning of Invariance. In *Proceedings of IEEE International Conference on Image Processing* (Austin, Texas, 1994).

[14] Weickert, J. Theoretical foundations of anisotropic diffusion in image processing. In *Computing, Suppl. 11*. 1996, pp. 221–236.

[15] Weickert, J. A review of nonlinear diffusion filtering. In *Scale-Space Theory in Computer Vision* (1997), B. ter Haar Romeny, L. Florack, J. Koenderink, and M. Viergever, Eds., vol. 1252 of *LNCS*, Springer, Berlin, pp. 260–271.

[16] Weickert, J. Examples of nonlinear diffusion filtering. `http://www.mia.uni-saarland.de/weickert/demos.html`, February 1998. (Accessed 4 Sep 2002).