A facial feature tracker for human-computer interaction based on 3D TOF cameras

Martin Böhme*, Martin Haker, Thomas Martinetz, and Erhardt Barth

Institute for Neuro- and Bioinformatics
University of Lübeck, Germany
Ratzeburger Allee 160
D-23538 Lübeck, Germany
E-mail: {boehme, haker, martinetz, barth}@inb.uni-luebeck.de
*Corresponding author

Abstract: We describe a facial feature tracker based on the combined range and amplitude data provided by a 3D time-of-flight camera. We use this tracker to implement a head mouse, an alternative input device for people who have limited use of their hands.

The facial feature tracker is based on geometric features that are related to the intrinsic dimensionality of multidimensional signals. We show how the position of the nose in the image can be determined robustly using a very simple bounding-box classifier, trained on a set of labelled sample images. Despite its simplicity, the classifier generalizes well to subjects that it was not trained on. An important result is that the combination of range and amplitude data dramatically improves robustness compared to a single type of data.

The tracker runs in real time at around 30 frames per second. We demonstrate its potential as an input device by using it to control Dasher, an alternative text input tool.

Keywords: 3D time-of-flight camera; facial feature tracking; head tracking; head mouse; geometric features; alternative and augmentative communication (AAC).

Reference to this paper should be made as follows: Böhme, M., Haker, M., Martinetz, T., and Barth, E. (2007) 'A facial feature tracker for human-computer interaction based on 3D TOF cameras', *Dynamic 3D Imaging* (Workshop in conjunction with DAGM 2007).

Biographical notes: Martin Böhme received his degree in computer science from the University of Lübeck in 2002. He spent six months at Kyoto University before returning to Lübeck as a research assistant. His research interests include 3D-TOF-based tracking, eye tracking, and gaze-contingent displays.

Martin Haker received his degree in computer science from the University of Lübeck in 2006 before joining the Institute for Neuro- and Bioinformatics (INB) as a research assistant. His research interests include 3D-TOF-based tracking and manifold learning.

Thomas Martinetz is director of the INB. He received his Ph.D. in neuroinformatics at the University of Illinois at Urbana-Champaign and has conducted research at the Neuroinformatics Research Center of Siemens

Copyright 2007 Inderscience Enterprise Ltd. Published in International Journal of Intelligent Systems Technologies and Applications, 5(3/4):264–273, 2008.
DOI: 10.1504/IJISTA.2008.021289.

AG in Munich and at the Institute for Neuroinformatics of the University of Bochum. The main lines of research at his institute are in biological information processing, in particular in neural vision, pattern recognition, and learning.
Erhardt Barth is currently a lecturer and senior scientist at the INB. His main research interests are in the areas of human and computer vision. He received his Ph.D. in electrical and communications engineering at the Technical University of Munich and has conducted research at the Universities of Melbourne and Munich, the Institute for Advanced Study in Berlin, the NASA Vision Science and Technology Group in California, and the Institute for Signal Processing in Lübeck. In May 2000 he received a Schloessmann Award.

1 Introduction

In the last decade, a novel type of image sensor – the 3D time-of-flight (TOF) camera – has been developed [5, 6, 8, 10]. This camera fuses the acquisition of both intensity and range data into a single device at a relatively low cost; the future pricing of such cameras is expected to be in the range of standard webcams. A 3D TOF camera works by emitting modulated infrared light; the phase shift of the reflected light provides a range measurement, and the amplitude, which depends on the reflectivity of the object, provides a conventional intensity image.
In this paper, we use a 3D TOF camera to implement a facial feature tracker and show how this tracker can be used as an alternative means of controlling a mouse cursor.
In contrast to conventional cameras, 3D TOF cameras have the obvious benefit of providing information about the three-dimensional structure of the scene. Computer vision applications that use this information can reasonably be expected to be more robust than if they used only a conventional intensity image, because geometric structure is typically more invariant than appearance.
Indeed, our tracker performs better on 3D TOF data than on intensity data alone. However, beyond this we have observed an interesting phenomenon in our results: It is not the range data by itself that improves the robustness of the results but rather the combination of range and intensity data; while both types of data perform roughly the same when used individually, the combination of the two yields substantially better results than either type of data alone. We intend to investigate whether this effect also occurs for other applications.
Previous work has already identified the nose as an important facial feature for tracking, see for example [3] and [11]. The first approach determines the location of the nose by template matching and gives very robust results under fixed lighting and at a fixed distance of the user from the camera. The second approach works by fitting a geometrical model of the nose to the image data.
We also consider the nose as being well suited for head tracking because, in terms of differential geometry, the tip of the nose is the point of maximal curvature on the surface of the face. Gaussian curvature would therefore seem to be a good choice as an image feature for tracking the nose, but it has the disadvantage that it is based on first and second order derivatives, which are sensitive to noise. We propose alternative features that can be related to generalized differential operators. In

previous work [4], we describe a nose detector based on these features that achieves good error rates using a simple threshold-based classifier.
The nose detector described previously [4] was a Matlab implementation that did not run in real time. In this paper, we describe a real-time C++ implementation of the nose detector, which we then use as the basis for a nose tracker. Because of the low error rate of the detector, the tracker requires only simple post-processing and tracking algorithms. We envisage that this type of tracker can be used for human-computer interaction and, particularly, for Augmentative and Alternative Communication. To demonstrate this type of application, we use the tracker to control the alternative text entry tool Dasher [9].

2 Geometric Features

To define the geometric features used in the classifier, we interpret the range and amplitude data as a particular type of surface, the *Monge patch* or *2-1/2-D* image, defined as a function $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^3, (x, y) \mapsto (x, y, f(x, y))$, where x and y specify a position on the image sensor and $f(x, y)$ is the corresponding range or amplitude value. (In practice, this function is sampled on an equispaced grid.)
We point out that the geometry of the range data patch is not identical to the geometry of the three-dimensional surface that was measured. This is because the Monge patch implies an orthographic projection, whereas the camera lens performs a perspective projection. However, in our application, the orthographic projection is a good approximation because the range variation within the face is small compared its distance from the camera (this is the so-called *weak perspective* assumption).
On this data model, we use a set of curvature measures (originally proposed in [1] and [12]) that provide basic and reliable alternatives to the Gaussian curvature K and the mean curvature H for the purpose of surface classification. We will briefly summarize the derivation.
Let us first recall the definition of Gaussian curvature for a Monge patch:

$$K = \frac{f_{xx} f_{yy} - f_{xy}^2}{(1 + f_x^2 + f_y^2)^2}. \tag{1}$$

In case only the sign of the curvature is relevant, one can rely on the DET-operator D, which can be formulated in terms of the determinant of the Hessian and is equal to the numerator of (1). Thus the DET-operator takes the following form:

$$D = f_{xx} f_{yy} - f_{xy}^2 = \det(h_{ij}) = d_1 d_2. \tag{2}$$

Here, d_1 and d_2 denote the eigenvalues of the Hessian. Rearranging the first part of the formula (see [1]) yields

$$D = \frac{1}{4}(f_{xx} + f_{yy})^2 - \frac{1}{4}(f_{xx} - f_{yy})^2 - f_{xy}^2 = (\Delta f)^2 - \epsilon^2, \tag{3}$$

where Δf denotes the Laplacian and ϵ is referred to as the eccentricity, which is defined as

$$\epsilon^2 = \frac{1}{4}(f_{xx} - f_{yy})^2 + f_{xy}^2. \tag{4}$$

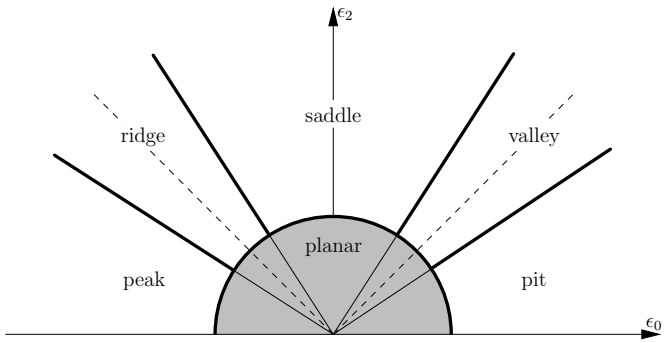


Figure 1 Discrimination of the six surface types *pit*, *peak*, *saddle*, *valley*, *ridge*, and *planar* within the feature space spanned by ϵ_0 (Δf) and ϵ_2 (ϵ).

The above formulation yields a relationship of the curvature to the Laplacian and the eccentricity. A generalized representation of the operators Δf and ϵ can be achieved in the Fourier domain by defining the generalized eccentricity ϵ_n via the following filter functions in polar coordinates ρ and θ, where $A(\rho)$ represents the radial filter tuning function:

$$C_n = \mathrm{i}^n A(\rho)\cos(n\theta), \quad S_n = \mathrm{i}^n A(\rho)\sin(n\theta). \tag{5}$$

Recall that the transfer functions of partial derivatives are of the form $(\mathrm{i}f_x)^n$ and $(\mathrm{i}f_y)^n$, where f_x and f_y represent the spatial frequencies and n denotes the order of differentiation. Even-order partial derivatives correspond to real transfer functions, whereas odd-order partial derivatives correspond to imaginary transfer functions. The transfer functions in (5) correspond to convolution kernels $c_n(x, y)$ and $s_n(x, y)$ in the image domain. Using these, we obtain the generalized eccentricity

$$\epsilon_n^2 = (c_n(x, y) * l(x, y))^2 + (s_n(x, y) * l(x, y))^2 \tag{6}$$

for $n = 0, 1, 2, \ldots$, which corresponds to $|\Delta f|$ for $n = 0$ and to the eccentricity ϵ for $n = 2$, when $A(\rho) = (2\pi\rho)^2$. The modulus of the gradient is defined by ϵ_n for $n = 1$ and $A(\rho) = 2\pi\rho$. In a purely geometrical interpretation, all measures ϵ_n are positive, and as a result one cannot distinguish between convex and concave curvature using ϵ_0 and ϵ_2. An extension to this formulation in [1] justifies the use of ϵ_0 with the sign of Δf, i.e. $\epsilon_0 = -c_0 * l$.

For practical applications, the radial filter tuning function $A(\rho)$ can be combined with a low-pass filter, e.g. Gaussian blurring of the form $G(\rho, \sigma) = \exp(-\pi\rho^2/4\sigma^2)$. Ideally, the low-pass filter should be adapted to the distribution of noise inherent in the data.

The measures ϵ_n for $n = 0$ and $n = 2$ can be used to distinguish between the six well-known surface types in the feature space spanned by ϵ_0 and ϵ_2. Figure 1 shows where the different surface types lie in feature space. Because the nose is a local minimum in the range data, we would expect the corresponding pixels to lie in the region labeled *pit*. Conversely, since the nose tends to be a local maximum in the intensity data, we would expect to find the corresponding pixels in the region labeled *peak*.

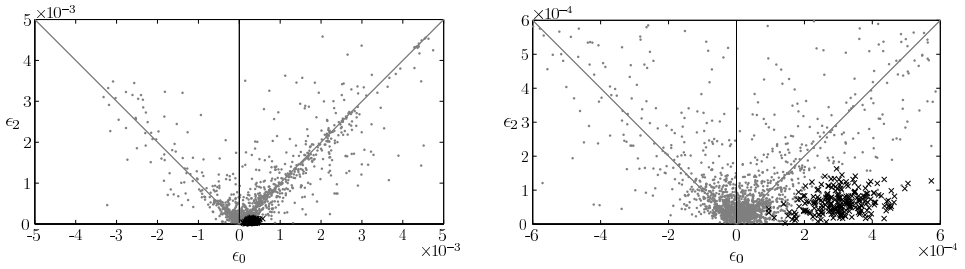


Figure 2 Left: Distribution of feature points for pixels taken from range data of the SR3000 camera projected into the 2D feature space spanned by ϵ_0 (Δf) and ϵ_2 (ϵ). Right: The feature space around the origin at a higher resolution. The black crosses represent feature points corresponding to the nose tip of various subjects and clearly cluster in the region associated with the surface type *pit* as expected. The grey dots represent randomly chosen non-nose pixels.

3 Nose Detection

To evaluate the potential of using the generalized eccentricity measures ϵ_0 and ϵ_2 to detect noses, we acquired a database of head images from different users with different head poses per user. We then hand-labelled the position of the nose in each image and compared the values of ϵ_0 and ϵ_2 at the labelled locations with the values at other, non-nose locations.

Figure 2 shows the results of this comparison; it plots the values of ϵ_0 and ϵ_2 on the range data. The values obtained at the labelled nose locations are plotted using black crosses, and the values at randomly chosen non-nose (and non-background) locations are plotted using gray dots.

As expected, the nose pixels cluster together in the *pit* region. Most of the non-nose pixels are fairly close to the origin (which corresponds to 0D, i.e. locally constant structures); most of the remaining pixels cluster around the diagonals (1D structures) with only a few pixels in the remaining regions (2D structures).

The results for the amplitude data are not shown because they are qualitatively similar, the main difference being that the nose pixels cluster together in the *peak* region; again, as expected.

The radial filter tuning function was set to $A(\rho) = (2\pi\rho)^2 \cdot \exp(-\pi\rho^2/(4\sigma^2))$ with $\sigma = 0.3$ for all feature computations. We expect that filter optimization will further improve the results.

Based on these results, we implemented a simple nose detector as follows: The radial structure of the feature space (see Figure 1) suggests that polar coordinates are a natural representation for points in the space. Therefore, we perform a conversion to polar coordinates r (radius) and ϕ (angle). We collect the feature values into a feature vector $\mathrm{F} = (\mathrm{F}_1, \ldots, \mathrm{F}_m)$, which contains the values of r and ϕ for the range and/or amplitude data, depending on which data we want to use for the classification. We then estimate the boundaries $\mathrm{F}_{\min}$ and $\mathrm{F}_{\max}$ of a bounding box enclosing the "nose" region in feature space. A pixel is classified as "nose" if $\mathrm{F}_{\min\, j} < \mathrm{F}_j < \mathrm{F}_{\max\, j}$ for all $j = 1, \ldots, m$ and "non-nose" otherwise.

The values for $\mathrm{F}_{\min}$ and $\mathrm{F}_{\max}$ are determined by taking a set of training images with hand-labelled nose locations and then choosing the bounding box that contains all of the training samples in feature space. The trade-off between detection rate and

false positive rate can be controlled by scaling the box around its centre; we obtain the new bounding box limits

$$\hat{\mathrm{F}}_{\min} = \mathrm{F}_{\mathrm{centre}} - \alpha\,\mathrm{F}_{\mathrm{halfwidth}}, \quad \hat{\mathrm{F}}_{\max} = \mathrm{F}_{\mathrm{centre}} + \alpha\,\mathrm{F}_{\mathrm{halfwidth}},$$

where $\mathrm{F}_{\mathrm{centre}} = \frac{\mathrm{F}_{\min}+\mathrm{F}_{\max}}{2}$ and $\mathrm{F}_{\mathrm{halfwidth}} = \frac{\mathrm{F}_{\max}-\mathrm{F}_{\min}}{2}$. We call α the "softness" parameter.

To avoid misclassifications in regions with low intensity and, thus, low measurement confidence, we used Otsu's method [7] to compute a threshold on the amplitude data and separate the foreground from the background. The background was set to a fixed value in both the range and the amplitude data.

This is obviously a very simple classifier, but despite its simplicity it yields very robust results, as we will show in Section 5.

4 Nose Tracking

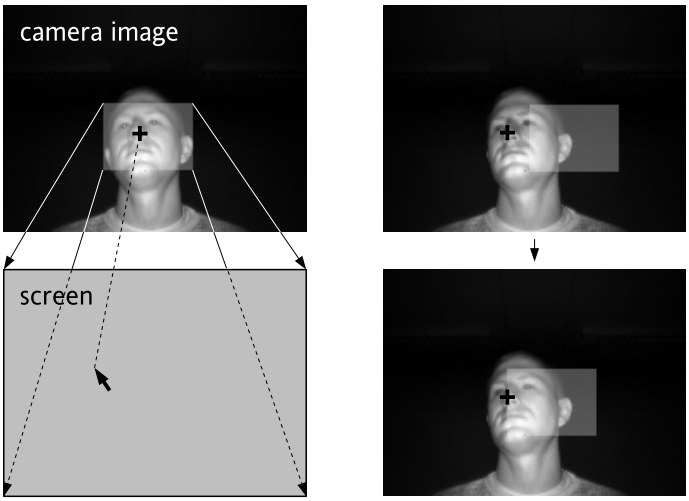
To track the nose and use it to control the position of the mouse pointer, we need to choose one of the potentially several nose locations that were detected, and we need to translate this nose location into a mouse cursor position.

First, we find all of the connected components in the image; each connected component yields one candidate location for the position of the nose, which we determine as follows: We first find the pixel whose feature vector is closest to the centre of the bounding box, i.e. for which $\|\mathrm{F} - \mathrm{F}_{\mathrm{centre}}\|_2$ is minimal. We then refine this location with subpixel precision by taking a small window around the candidate location and computing a weighted centroid of the pixels in the window, where the weight of a pixel depends on the distance of its feature vector to $\mathrm{F}_{\mathrm{centre}}$; specifically, the weight function is a Gaussian centred on $\mathrm{F}_{\mathrm{centre}}$.

Since the raw results delivered by the nose detector are already quite good (see Section 5), most camera frames contain only a single connected component and thus a single candidate location, placed correctly on the nose. If the current frame contains more than one candidate location, we choose the candidate that is closest to the position of the nose in the previous frame. For the very first frame, or if no nose was found in the previous frame, we simply choose one of the candidates arbitrarily. Even if we choose the wrong candidate, it is quite likely that the next frame will not contain any misdetections, and from that point on, we will be tracking the correct position. This is a simple strategy for eliminating misdetections, but it works quite well in practice.

Once the position of the nose has been identified, we need to convert it into a position on the screen. We define a rectangular "active region" in the camera image, sized so that when the user rotates his or her head, they are able to place their nose anywhere within this region. We then convert the nose position to a mouse cursor position by defining a mapping from camera image coordinates to screen coordinates such that the borders of the active region are mapped to the borders of the screen (see Figure 3a.) The mapping has to flip the horizontal axis (for clarity, this is not shown in the figure) because a head movement to the left causes the position of the nose in the camera image to move to the right.

Two questions remain to be solved: Where in the image should we place the active region? And what should be done if the user's seating position changes such that



a) b)

Figure 3 a) Illustration of the mapping between the position of the nose in the camera image and the position of the mouse cursor on the screen. The shaded rectangle is the active region, the cross marks the position of the nose, and the arrow symbolizes the mouse cursor. For clarity, the horizontal flip contained in the mapping is not shown. b) Adjustment made when the nose moves outside the active region. Top: Before the adjustment of the active region. Bottom: After the adjustment.

the nose leaves the active region? We solve these problems using an implicit calibration procedure that continually adjusts the position of the active region. For example, if the nose moves beyond the left border of the active region, we shift the active region to the left until the nose is just inside the active region again (see Figure 3b). Effectively, users "drag" the active region along as they change their seating position. This procedure works equally well for setting the initial position of the active region: A quick rotation of the head left, right, up, and down is sufficient to put the active region in a suitable position. If desired, the size and initial position of the active region could also be determined during an explicit calibration phase.

5 Results

The algorithms were implemented in GNU C++ and run on a 2.8 GHz Pentium 4 under Linux 2.6. The FFTW library [2] was used for the FFT transforms needed to calculate the generalized eccentricity measures. Images were acquired using a MESA SR3000 camera [5], which has a resolution of 176 by 144 pixels.

To evaluate the error rate of the nose detector, we first tested it on a set of static images, consisting of images from 13 subjects, with nine head orientations per subject. The position of the nose was hand-labelled in each image. The data from three subjects was used to estimate the bounding box for the nose detector, which was then tested on the remaining ten subjects. We calculated the detection rate as the percentage of images in which the nose was identified correctly (to within

Figure 4 Examples of detection results. The amplitude data (left column) and the range data (right column) are given for four subjects. All pixels identified as nose pixels by our detector are marked in each image, the cross simply highlighting the locations.

five pixels distance from the hand-labelled position) and the false positive rate as the percentage of images where at least one non-nose pixel was falsely classified as "nose". We adjusted the softness parameter until the false negative rate (one minus the detection rate) and the false positive rate were equal, thus obtaining an equal error rate (EER). When nose detection was performed on the range and amplitude data individually, we obtained EERs of 0.64 and 0.42, respectively, When nose detection was performed on the combination of both types of data, we obtained an EER of 0.03 (see [4] for details). Figure 4 shows some examples of detection results. The softness parameter that corresponded to the EER was chosen to set the bounding box for the nose tracker. On the combined range and amplitude data, the tracker ran at a rate of 27 frames per second. This is adequate for the frame rate we run the camera at but could be improved further (for example, we currently use a complex-to-complex FFT for simplicity of implementation, which could be replaced with a real-to-complex FFT). A video showing the nose tracker in action is available on the web at `http://www.inb.uni-luebeck.de/~boehme/nosetrack_demo.mp4`.
To evaluate the usefulness of the nose tracker for interaction tasks, we used it to control the alternative text input tool Dasher [9]. Dasher can be controlled using a variety of pointing devices, which are used to "steer" towards the letters to be input. A test subject using the nose tracker with Dasher to input text from an English language novel achieved 12 words per minute (wpm). For comparison, the rate that can be achieved using an eye tracker is between 15 and 25 wpm [9].

6 Conclusions

We have demonstrated how a very simple classifier based on geometric features can be used to detect a user's nose robustly in combined range and amplitude data obtained using a 3D TOF camera. A particularly interesting result is that markedly better results were obtained on the combination of range and amplitude data than on either type of data alone.
Based on this classifier, we implemented a real-time nose tracker. To demonstrate the usefulness of this tracker for human-computer interaction, we have shown that

it can be used effectively to control the alternative text entry tool Dasher.

Acknowledgments

This work was performed within the ARTTS project, which is funded by the European Commission (contract no. IST-34107) within the Information Society Technologies (IST) priority of the 6th Framework Programme. This publication reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

References and Notes

1 Erhardt Barth, Terry Caelli, and Christoph Zetzsche. Image encoding, labeling, and reconstruction from differential geometry. *CVGIP: Graphical Models and Image Processing*, 55(6):428–46, November 1993.

2 Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW. *Proceedings of the IEEE*, 93(2):216–231, 2005.

3 Dmitry O. Gorodnichy. On importance of nose for face tracking. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition (FG'2002)*, pages 188–196, Washington, D.C., May 2002.

4 Martin Haker, Martin Böhme, Thomas Martinetz, and Erhardt Barth. Geometric invariants for facial feature tracking with 3D TOF cameras. In *Proceedings of the IEEE International Symposium on Signals, Circuits & Systems (ISSCS)*, volume 1, pages 109–112, Iasi, Romania, 2007.

5 Thierry Oggier, Bernhard Büttgen, Felix Lustenberger, Guido Becker, Björn Rüegg, and Agathe Hodac. SwissRanger^{TM} SR3000 and first experiences based on miniaturized 3D-TOF cameras. In *Proceedings of the 1st Range Imaging Research Day*, pages 97–108, Zürich, Switzerland, 2005.

6 Thierry Oggier, Michael Lehmann, Rolf Kaufmann, Matthias Schweizer, Michael Richter, Peter Metzler, Graham Lang, Felix Lustenberger, and Nicolas Blanc. An all-solid-state optical range camera for 3D real-time imaging with sub-centimeter depth resolution (SwissRanger). In *Proceedings of SPIE*, volume 5249, pages 534–545, 2004.

7 Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, January 1979.

8 Thomas Spirig, Peter Seitz, Oliver Vietze, and Friedrich Heitger. The lock-in CCD – two-dimensional synchronous detection of light. *IEEE Journal of Quantum Electronics*, 31(9):1705–1708, 1995.

9 David J. Ward and David J.C. MacKay. Fast hands-free writing by gaze direction. *Nature*, 418(6900):838, 2002.

10 Zhanping Xu, Rudolf Schwarte, Horst Heinol, Bernd Buxbaum, and Thorsten Ringbeck. Smart pixel – photonic mixer device (PMD). In *Proceedings of the International Conference on Mechatronics and Machine Vision in Practice*, pages 259–264, Nanjing, China, 1998.

11 Lijun Yin and Anup Basu. Nose shape estimation and tracking for model-based coding. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01)*, volume 3, pages 1477–1480, May 2001.

12 Christoph Zetzsche and Erhardt Barth. Fundamental limits of linear filters in the visual processing of two-dimensional signals. *Vision Research*, 30:1111–1117, 1990.