Analog VLSI Motion Sensors

Thesis by
Rainer A. Deutschmann
July 29, 1997

Technische Universität München/Germany
California Institute of Technology/USA

Rainer A. Deutschmann
email: contact@rainer-deutschmann.de
www.rainer-deutschmann.de

Abstract

Recent developments in analog VLSI technology and circuit design provided new impact to solving problems through analog computation. One particularly challenging task is to build smart vision sensors, that not only *see*, but to some degree also *understand*. In this work three highly integrated sensors have been developed, which extract motion information from a visual scene through analog parallel real time computation. The sensors have been implemented in standard CMOS processes and are fully functional. Their use in technical applications and as front end for biologically inspired high level vision systems is suggested.

The ‘CMotion1d sensor’ is the first working sensor to implement the gradient method for velocity computation in 1-D. The sensor operates reliably over a wide range of speeds. The ‘Gradient2d sensor’ uses a different gradient based algorithm to compute a 2-D motion vector field. The combination of small pixel size in a 2-D implementation, desirable motion output for 2-D stimuli and high sensitivity makes this sensor exceptional. Finally the ‘FTC sensor’ implements a new feature based algorithm to compute the direction-of-motion flow field in real time.

Contents

List of Figures

Chapter 1

Introduction

Who has ever tried to implement a vision system to function in a real world situation can only be fascinated by how well biological systems perform. Effortless a fly escapes our attempt to catch it, equally effortless a bird can catch the same fly while flying at high speeds. A hawk can spot its prey over an incredible distance. While walking we can read a book, coming from a dim room into plain sunlight we can go on reading, and although concentrating on the book we don't run into a wall. Given equal size and power consumption constraints, not a single artificial system comes even close to perform this well.

Before 1980 computer vision research has mainly been focussed on static images. More recently computer vision algorithm have been developed that can deal with image motion. Taking not only two spatial but also the temporal dimension into account proved to be challenging theoretically as well as for implementations. Traditionally artificial vision systems consist of a camera and a computer, on which the acquired image sequence is processed. For 'prototyping' sometimes the input image sequence is even recorded and replayed in slow motion. Few systems are known that perform within the time scale of real world tasks [Dic95]. The required computing power in these cases is enormous.

Very recently powerful tools became available that could make for the first time highly integrated and parallel vision system feasible. Artificial vision systems could come a little closer to the exceptional performance of biological systems. Rapid improvements in chip technology made it possible to custom design very large scale integrated (VLSI) circuits for a moderate price. In a step away from digital systems analog computation was rediscovered as effective method for some problems and suitable for implementation on commercial chip processes. Key elements for a vision system such as photoreceptors and elementary functions were developed.

1.1 About this Thesis

The work reported in this thesis has been done in probably the most exciting time for anybody interested in integrated vision systems. Smart sensors, which include photosensing and computing circuits on one single chip, had previously been proven to work for pure spatial processing, and first steps towards integrated motion and velocity sensors, mostly in 1-D, had been taken.

The focus of this work are analog VLSI motion sensors. Three integrated sensors have been developed and are fully functional. Every step from development of the motion algorithm and circuit to the layout of the physical masks for the chip fabrication and eventually the sensor testing was done by the author. The sensors are intended to be used as a real time front end for a high level vision system.

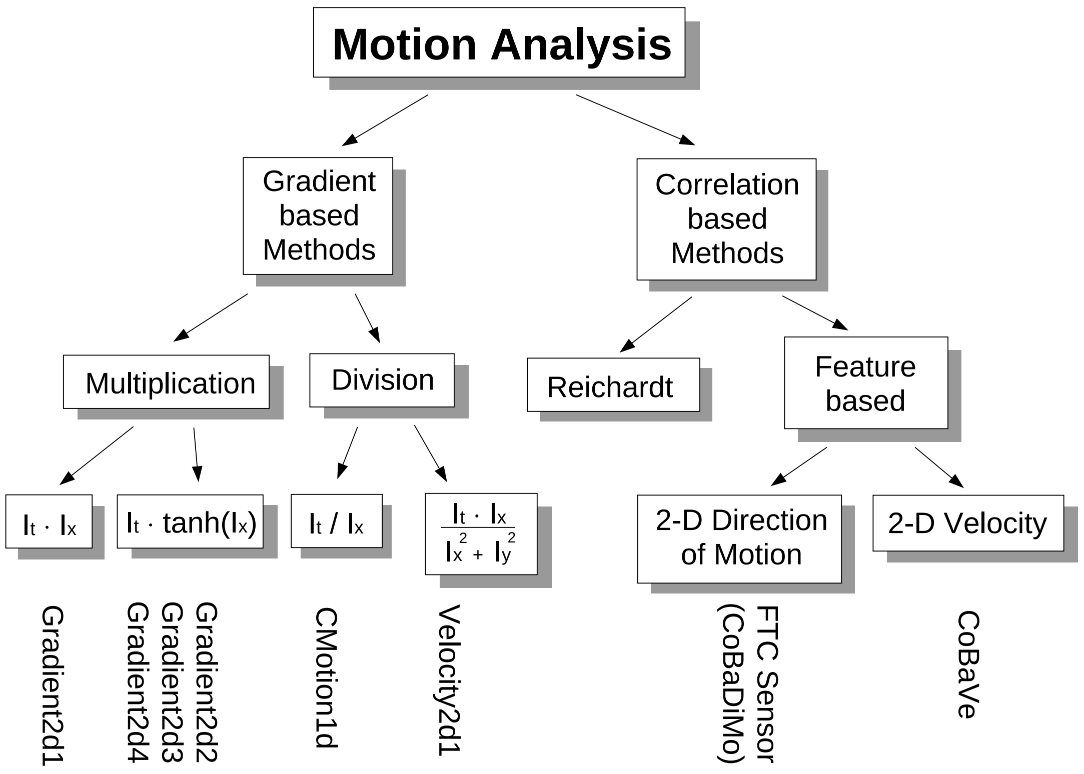


Figure 1.1: Classification of motion processing schemes. The sensors that are presented in this work are indicated on the bottom. Experimental results from Gradient2d2, CMotion1d and the FTC sensor will be shown.

In Figure 1.1 a possible classification of motion processing schemes is shown. On the one side *gradient based methods* use temporal and spatial derivatives to compute a motion signal, which can be a vector quantity but not necessarily has to be proportional to the stimulus velocity. Gradient based methods can further be split up into multiplication and division based methods. Instances of the former are the product of the temporal and the spatial derivative, and the product of temporal derivative and the tanh function of the spatial derivative. When both derivatives are divided in a proper manner, true stimulus velocity can be reported. This case will be called throughout this thesis *gradient method*.

One the other side *correlation based methods* can be split up into the Reichardt type methods, where in a strict sense light intensities at different times and pixels are correlated by means of multiplication, and methods that track features (e.g. edges or more complex tokens) for detecting motion.

Some of the methods, like the gradient method and some feature based methods, produce a velocity output, whereas others, like the multiplication schemes, provide a motion signal which is dependent on more parameters than the stimulus velocity.

For this work instances of sensors to every class except for the Reichardt type have been developed. The CMotion1d sensor is an implementation of the true gradient method for 1-D velocity. The Gradient2d sensor family performs a multiplication of temporal and spatial derivatives on a 2-D pixel array. The FTC sensor belongs to the class of feature based sensors, it computes the direction-of-motion vector field on a 2-D array based on temporal edge detection.

1.2 Organization of this Thesis

This thesis consists of one introductory chapter and three main chapters, in which the sensors are presented. The thesis ends with a concluding chapter, and a German summary.

- In **Chapter 2** a short introduction to analog VLSI is given and the process of building analog VLSI chips is outlined. Part of the future of analog VLSI will depend on the question, how well sensors can be implemented in larger arrays. This topic is treated in one section. The following two sections focus more specifically on motion sensors and the sensors presented in this thesis. First previous work in the field is reviewed and critical points are emphasized. Second some remarks are made that are necessary for the understanding of the subsequent chapters where the sensors are presented.

- In **Chapter 3** the CMotion1d sensor is presented. The name derives from the fact, that the sensor can compute circular motion with a 1-D array of elementary motion detectors (pixels). The main focus of this chapter, though, lies on the characterization of the pixel. The CMotion1d sensor is the first working implementation of the gradient method for velocity computation in 1-D. The chapter begins with explaining, why implementing the gradient method is difficult. Next the actual implementation is described and circuit specifics are given. A section is dedicated to discussing the fundamental limits of the gradient method for velocity computation, when implemented in analog VLSI. In five sections experimental results from the CMotion1d pixel are presented. Sample outputs are shown and the dependency of the motion output on stimulus velocity, spatial frequency, contrast and orientation are investigated. Subsequently the performance of the CMotion1d sensor for computing rotational velocity is demonstrated. An implementation of the gradient method for velocity computation in 2-D has been built based on the experience gained from the CMotion1d sensor. Although results from this sensor have not yet been collected, the sensor is presented in the next section to demonstrate that despite the complexity of the required computation an implementation in analog VLSI is possible. The chapter concludes with a summary and discussion.

- In **Chapter 4** the Gradient2d sensor is presented. This is a 2-D sensor which implements a multiplication based gradient scheme and was designed with focus on the smallest possible pixel size for a motion sensor. A version of the Gradient2d sensor is described which consists of a 7×7 pixel array. The chapter starts off which a description of the circuit implementation.

In the five following sections results from the Gradient2d sensor are demonstrated: Sample outputs show the time course of the motion output, and the motion output dependence on stimulus velocity, orientation, contrast and spatial frequency is shown. In a separate section the Gradient2d sensor is tested for very low contrasts. The chapter concludes with a summary and a preview on future work.

- In **Chapter 5** the Facilitate, Trigger and Compare (FTC) sensor is presented. This sensor computes the direction-of-motion vector field on a 12×13 pixel array with a new feature based algorithm. After an introduction the FTC algorithm and its implementation are described. Then, as with the other sensors, the output of the elementary motion detector is shown. In the two following sections the contrast and velocity sensitivity is investigated as well as the sensor response for stimuli of different orientations. In order to give some intuition of the real time sensor performance, which is difficult to do in a printed medium, the sensor output has been captured over time at high frame rates. Sample image sequences are presented. In the two following sections remarks on the bias settings and the scanning speed are made. Next sensor limitations are discussed in detail. The chapter concludes with a summary and ideas for future work.
- **Chapter 6** concludes this thesis and
- **Chapter 7** summarizes the thesis in German language.

Chapter 2

Analog VLSI Sensors for Motion Processing

2.1 An Analog VLSI Primer

Very Large Scale Integrated (VLSI) circuits are densely packed and interconnected semiconductor devices on a single chip, in contrast to circuits made up from discrete components. Todays information and communication technologies rely on VLSI chips for fast and reliable computation. The vast majority of VLSI chips is based on silicon. The physical materials for VLSI circuits are layers of mono crystalline silicon (for the substrate), silicon oxide (isolating the conducting layers), conducting polysilicon 'poly' (for gates and interconnects) and several metal layers. The substrate is locally doped with foreign atoms, so that two types of transistors are obtained: nMOS and pMOS transistors are the key devices for VLSI circuits. These transistors are three terminal devices, where a voltage on the gate controls nonlinearly the current between source and drain. Because the electric field between the gate and source-drain region is used, these transistor are called Field Effect Transistors (FETs). The circuit symbol for nFETs and pFETs is shown in Figure 2.1. Complementary Metal Oxide Semiconductor (CMOS) circuits consist of both nMOS and pMOS transistors. Other devices can be built in CMOS technology: MOS capacitors with Source and Drain of a MOS transistor tied together, floating capacitors between poly layers or between a poly and a metal layer, photo diodes and photo transistors, and resistors of low resistance.

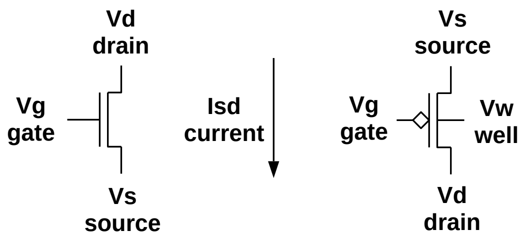


Figure 2.1: Circuit symbols for nMOS and pMOS transistor.

Analog VLSI in contrast to *digital* VLSI means, that the transistors are not used only as on-off switches, but the entire intermediate range of operation is used for computation. Depending on the gate voltage two regimes of transistor operation can be distinguished. For gate voltages not further than about 0.7 V away from the source voltage the transistor is subthreshold, the source-drain current is exponentially dependent on the gate voltage and its magnitude is smaller than a few hundred nano Amperes. For larger gate voltages the dependency is quadratic. Most of the circuits presented in this thesis are operated in the subthreshold regime, for computational and power consumption reasons.

In the subthreshold region the physical process of current conduction is diffusion, as compared to drift for above threshold operation. It therefore is intuitively clear, that the dependence of the source-drain current I_{sd} on the gate voltage V_g, the source voltage V_s and the drain voltage V_d is exponential. The following equations for nMOS and pMOS transistors are found from a device analysis:

$$\begin{aligned} I^n_{sd} &= I_o\, e^{\frac{\kappa\, V_g}{V_T}} \left(e^{-\frac{V_s}{V_T}} - e^{-\frac{V_d}{V_T}} \right) \\ I^p_{sd} &= I_o\, e^{\frac{\kappa\,(V_w - V_g)}{V_T}} \left(e^{-\frac{V_w - V_s}{V_T}} - e^{-\frac{V_w - V_d}{V_T}} \right) \end{aligned} \tag{2.1}$$

All voltages are referenced with respect to the substrate. V_s, V_d and V_g are source, drain and gate voltage, respectively. V_w is the well voltage for pFETs, which usually is tied to Vdd. I_o depends on the transistor geometry and process parameters of the chip fabrication. κ is the back gate coefficient and accounts for the fact, that the influence of the gate on the channel is diminished by the depletion region-substrate capacitance. $V_T = e/k\,T$ is the thermal voltage at temperature T.

In many practical cases the source voltages are at GND or Vdd, and the drain voltages are more than 100 mV away from the source voltage. In this case equation 2.1 simplifies to

$$\begin{aligned} I^n_{sd} &= I_o\, e^{\frac{\kappa\, V_g}{V_T}} \\ I^p_{sd} &= I_o\, e^{\frac{\kappa\,(V_w - V_g)}{V_T}} . \end{aligned} \tag{2.2}$$

The transistor is called ‘in saturation‘, the source-drain current is to first order exponentially dependent only on the gate voltage and independent of source and drain voltages.

For a more detailed introduction to semiconductor physics the reader is directed to the book by Sze [Sze81]. More on analog VLSI circuits and design can be found in the book by Weste and Eshraghian [WE85], the book by Gray and Meyer [GM93] and the book by Carver Mead [Mea89].

2.2 Analog Computation

In the last section it was mentioned that unlike in digital chips, where the transistors are used just as switches, in analog VLSI systems the entire physics of the devices is used. This allows for a very compact representation of the signals, which are continuously valued currents and voltages. One signal is transmitted on one wire. The fundamental computational primitives are adding and subtracting currents, taking the logarithm of currents and exponentiating voltages. Based on these primitives more complex operations can be implemented, a few of which shall be mentioned here:

- almost arbitrary power laws, such as multiplication, division and squaring
- functions involving exponentials, such as the tanh function
- copying of currents with only two transistors, arbitrary gain factors can be introduced
- integration and differentiation
- absolute value
- spatio-temporal filters
- sample and hold

All of the above functions can be implemented with just a few transistors and capacitors. In summary analog VLSI not only allows for a high device integration density, but also for a very compact implementation of complex functions. This is the reason why analog VLSI technology can be successfully used to build parallel systems.

2.3 The Process of Chip Design

Nowadays digital chip design has become very automated. Usually the chip designer works at a higher level than logical gates or even single transistors. This is because the transistor is used as switch, the exact transistor geometry and location do not usually have to be defined manually.

The analog VLSI designer, in contrary, cares very well about transistor geometry and location. Some transistors have to match better than others, so they are made larger. Some have to be strong, so they are laid out wide, whereas if a small Early voltage is desired, the transistor has to be long. The designer then has to be careful, because larger transistors have larger parasitic capacitance. Low parasitic capacitance on a node is achieved by bringing the involved devices close together. If the top metal layer, which is used as a light shield, has to be broken up, then this should be avoided above transistors that shall have low leakage currents. If pixel with mixed signals are laid out, as are all the pixels presented in this work (digital and analog signals are processed in close neighborhood), then capacitive coupling between digital and analog wires can be avoided through clever routing techniques. These are just a few examples of why the analog VLSI designer has to work on the lowest level of layout, which means the layout of the physical masks of the semiconductor process.

In Figure 2.2 the actual process of designing a VLSI chip is shown. From the algorithm, which should be designed with regard to a VLSI implementation, a circuit is worked out. This is done with the help of software simulations of parts or the entire circuit. The circuit is cast into a layout, which can involve up to weeks for a complicated pixel. If design rules are violated, the chip can be malfunctioning. A design rule checker (DRC) can help finding design rule errors. At the same time the circuit is specially prepared in a schematic editor, so that both layout and schematic can eventually be compared, which is done by a special computer program (LVS). If no design rule errors are found and the layout seems to match the intended schematic (or the designer is running out of time before the chip deadline), the layout is saved in a special format and submitted to the chip fabrication facility (fab), which can conveniently be done through email. After a period of two

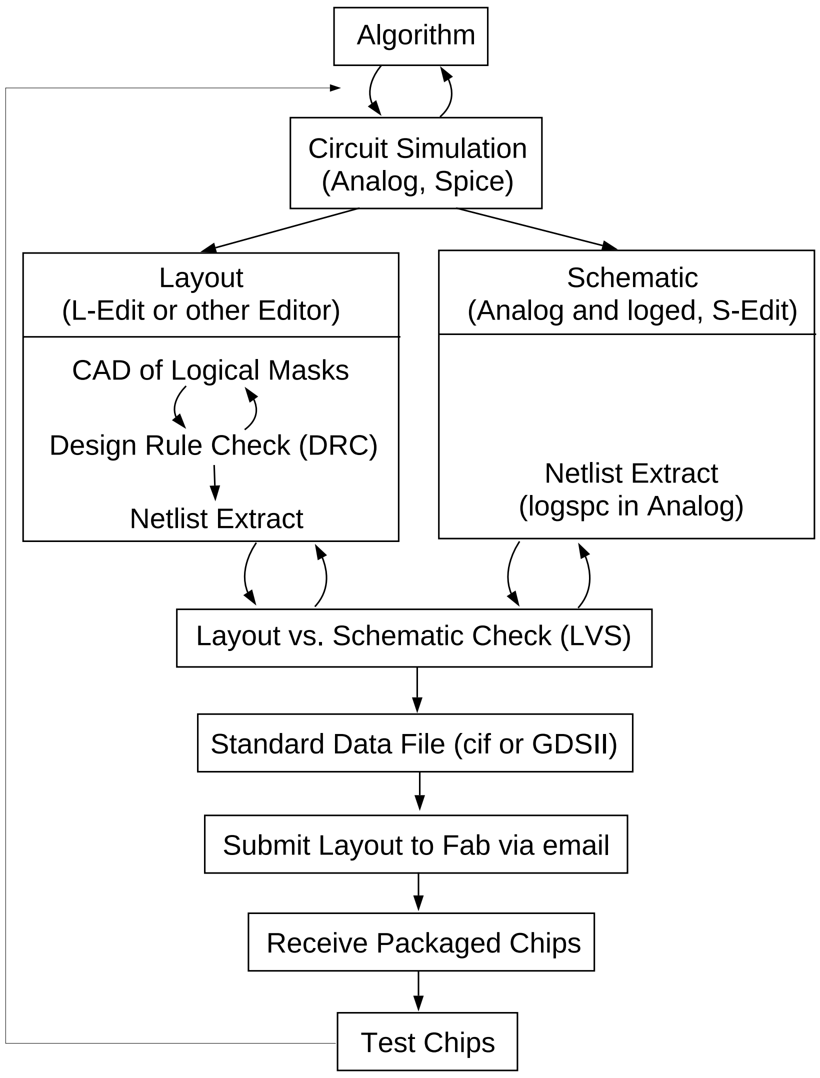

Algorithm
Circuit Simulation (Analog, Spice)
Layout (L-Edit or other Editor)
CAD of Logical Masks
Design Rule Check (DRC)
Netlist Extract
Schematic (Analog and loged, S-Edit)
Netlist Extract (logspc in Analog)
Layout vs. Schematic Check (LVS)
Standard Data File (cif or GDSII)
Submit Layout to Fab via email
Receive Packaged Chips
Test Chips

Figure 2.2: What is takes to design a chip.

months (if the designer is lucky), the fabricated and readily packaged chips are received and can be tested.

2.4 Sensor Architecture and Scalability in Size

2.4.1 Parallel systems with serial read-out

The sensors presented in this work all have a true parallel architecture. From the photo-detection stage on all computation is performed in parallel in every single pixel, and the output of every pixel is available at all times. A vision system with parallel architecture and nearest neighbor interaction has the great advantage that its resolution can be increased without any change to the algorithm.

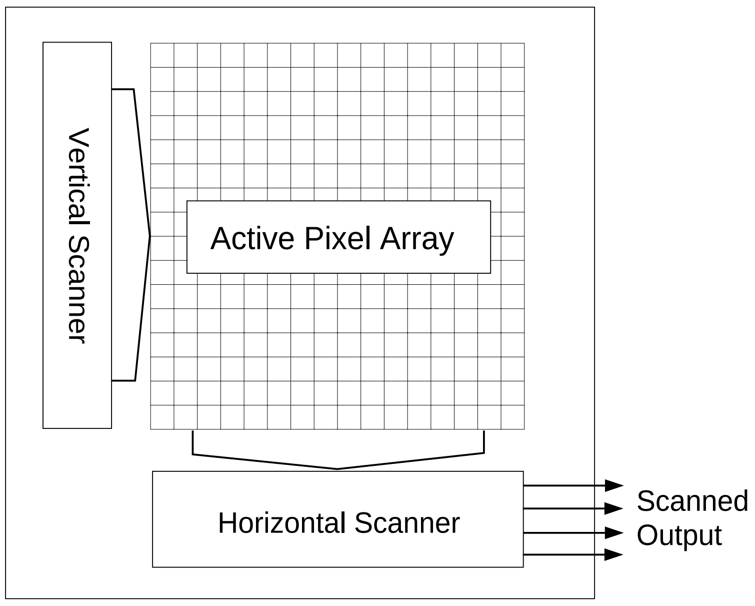


Figure 2.3: Diagram of a scanned parallel sensor system.

More specifically in the case of real time parallel systems the real time performance is maintained. The parallel architecture of the sensors presented here is possible because of the compact implementation of the computation in analog and hybrid VLSI. Due to the true parallel implementation of the sensors, the resolution in a 1-D implementation scales linearly with the chip dye size and the resolution in a 2-D implementation scales quadratically with the chip dye size. In fact, the scaling to larger chip dye sizes is even more favorable, because the digital support circuitry and the bonding pads stay the same physical size and thus take less percentage of the chip area for larger chips. On top of that instead of using all four sides for bonding pads only three or two sides need to be used such that even more chip area can be dedicated to the pixel array.

Even though the sensors are truely parallel, one serial component has to be used in every implementation where local information is to be accessed: the circuitry for communication with the external world. For any sensor with resolution larger than 5 by 5 the number of pins on the chip (usually 40 on a standard DIP40 package) is not sufficient for communication. There are at least two ways to deal with this bottleneck. The so called Address Event Protocol [Boa96a] [Boa96b] [Boa96c] can be used for asynchronous communication, or serial scanners can be used to serially clock through the array and probe every single pixel step by step. In the work presented here serial scanners are used. A detailed description is given in section 2.6.1. Serial scanners are a concern for scaling. Even if the chip size due to the parallel architecture can be increased with every single pixels still computing in real time, the temporal sampling rate through the serial scanner will decrease for larger sensors for a given maximal clock rate. From the outside the chip then just looks like a serial system, see figure 2.3.

For the FTC sensor a frame rate of 40 Hz could be achieved, which by itself beats the performance of CCD based systems, where the maximal frame rate usually is 30 Hz. For a larger version of the FTC sensor and the same type of scanner the frame rate would decrease. It has to be said, though, that the main focus of this work laid on the development of the sensor system, and no work has been done to improve the scanners. It is known to the author that much more sophisticated scanners are implementable and much higher frame rates can be achieved even for high resolution sensors.

2.4.2 Parallel systems with real-time read-out

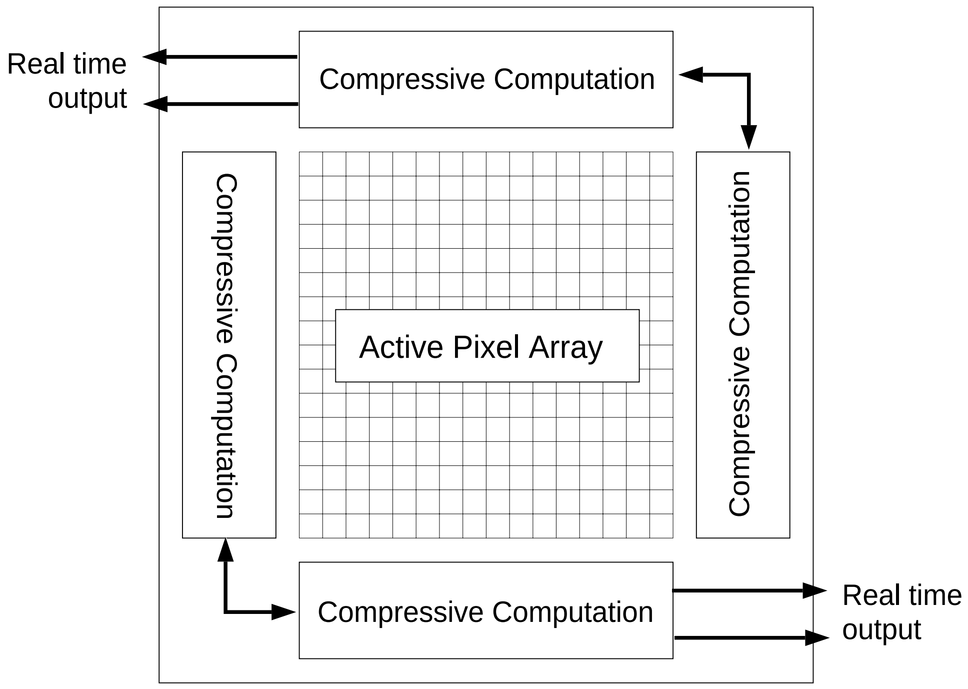


Figure 2.4: Diagram of a fully parallel sensor system.

In the case of parallel sensor systems where not the local information is to be obtained, but only few parameters such as the global direction-of-motion or the location of the focus of expansion are computed, these parameters can be read off the chip in real time because in this case no scanners are needed. This type of parallel system is truely scalable. Figure 2.4 shows the general architecture of this kind of sensor. The information is precomputed in parallel, then fed into circuitry adjacent to the array where the 2-D information is compressed into several 1-D streams. The output of this circuitry (compressive computation) are several 0-D parameters. These parameters need not to be scanned off the chip but rather can be passed out at once.

Even sensor systems with serial scanners can be used as parallel systems with real-time readout. As described in section 2.6.1 the serial scanners that are used for the sensors of this work can be driven in a special way to sum up the results of all pixels or a subclass of them. This corresponds to averaging. This average can be read out continuously in time without scanning through the array. In chapter 3 results for the average velocity of rotation for the CMotion1d sensor are shown.

2.4.3 Systems with parallel and serial computation

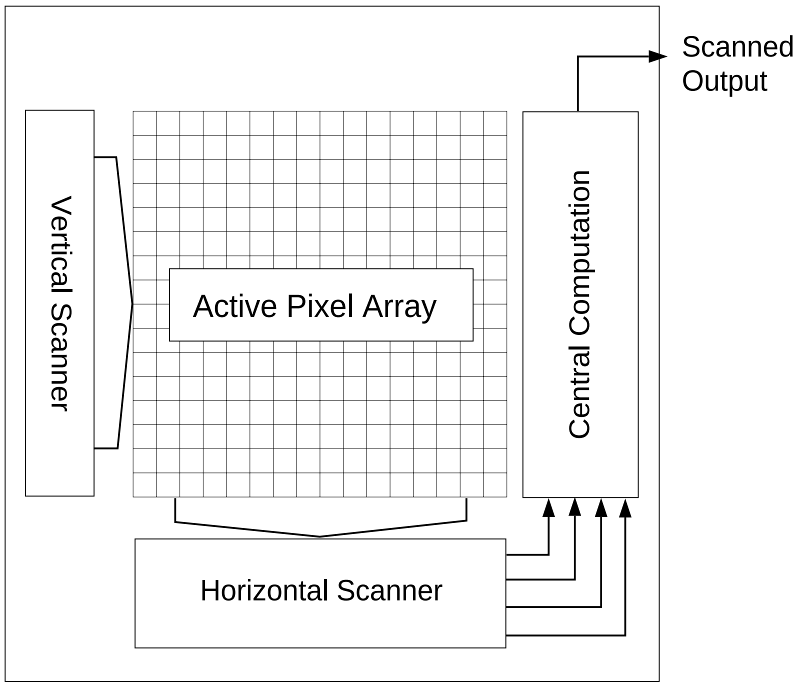


Figure 2.5: Diagram of a parallel/serial sensor system.

The gradient based velocity sensor Velocity2d presented in section 3.12 is an example for an implementation of a fairly complex computation. Because of the many computational steps involved for the velocity computation in 2-D, the pixels are the largest of all sensors presented in this work. Considering that the parallel computation will be broken up into a serial readout due to the serial scanner anyways, a new scheme of computation suitable for implementation in VLSI for complex sensors is suggested here (see Figure 2.5 and section 3.12). Only those computations which cannot be shared are performed locally at every pixel. All other computation is performed centrally on the sensor. The circuitry for the central computation exists only once and receives input from the pixel that is currently selected by the scanner. The output of the central circuitry is read off the chip.

2.4.4 Scaling and device mismatch

Due to the fabrication process of integrated circuits substantial variations of device parameters exist across the wafer (see for example [PAW94]). These variations tend to have space constants on the order of millimeters. Effects of these consistent device mismatches are observed even on 'tiny chips' (2.2mm$\times$2.2mm), but chips with large dye sizes are affected more seriously. Scaling to large chip sizes for very sensitive circuits can therefore be problematic. For example feature based motion sensors, like the FTC sensor presented in this work (see chapter 5), rely crucially on the temporal edge detector. If there were differences in the threshold of spike generation for different parts of the sensor, then for a given stimulus one part of the sensor would operate correctly but the other part would fail.

There are ways to ameliorate this problem:

- Adaptive circuits can set their proper operating point automatically. Slight variations in device parameters can be compensated for. Adaptation can be implemented with feedback systems, adaptive elements and floating gate technology. The downside of locally adaptive circuits in pixel array is their space consumption and in the case of floating gate technology the requirement of high voltages on the chip.

- Different bias voltages at different parts of the chip can be used for the same type of bias. Instead of only using one bias wire to all pixels for critical biases the chip can be supplied with different versions of the same bias such that device variations across the chip can be compensated for.

2.4.5 The price for scaling to larger chip sizes

As an example the FTC sensor presented in chapter 5 has a resolution of 13×12 pixels on a 2.2mm$\times$2.2mm chip. The pixel size is 119μm$\times128\mu$m, the technology is 1.2μm. On a 9.4mm$\times$9.7mm chip, which is available through MOSIS for US$ 17.490, a resolution of 75×72 can be obtained. Table 2.6 shows possible pixel resolutions for some of the prototype chips developed in this work. Dye sizes and respective prices were obtained from the MOSIS web page [MOS]. Figure 2.7 demonstrates how the chip prize scales almost linearly with the chip area. Even though for small chip sizes the price per chip for the 2.0μm technology is cheaper, for larger chip sizes the 1.2μm process becomes favorable. The comparison of chip technologies based on chip area is only useful if the chip is used for test circuitry which does not require a high resolution array. If high resolution arrays are demanded, then one has to compare the chip technologies in terms of how much circuitry can be implemented on a given chip area. In this case the lambda measure is useful. In this comparison for any chip size the 1.2μm technology is cheaper.

Sensor Name	Gradient2d2	Gradient2d4	FTC (CoBaDiMo)	CoBaVe
technology	2.0 µm	1.2 µm	1.2 µm	1.2 µm
pixel size	217µm · 201µm	112µm · 112µm	119µm · 128µm	105µm · 115µm
dye size	2.2mm · 2.2mm	2.2mm · 2.2mm	2.2mm · 2.2mm	2.2mm · 2.2mm
resolution	7 · 7	15 · 15	13 · 12	14 · 17
price [US$]	680	1000	1000	1000
dye size	6.9mm · 6.8mm	4.6mm · 4.7mm	4.6mm · 4.7mm	4.6mm · 4.7mm
resolution	29 · 31	36 · 37	34 · 33	39 · 36
price [US$]	7480	4320	4320	4320
dye size	7.9mm · 9.2mm	9.4mm · 9.7mm	9.4mm · 9.7mm	9.4mm · 9.7mm
resolution	34 · 43	79 · 82	75 · 72	85 · 80
price [US$]	14870	17490	17490	17490

Figure 2.6: Scaling of price and resolution with chip size for some of the sensors presented in this work.

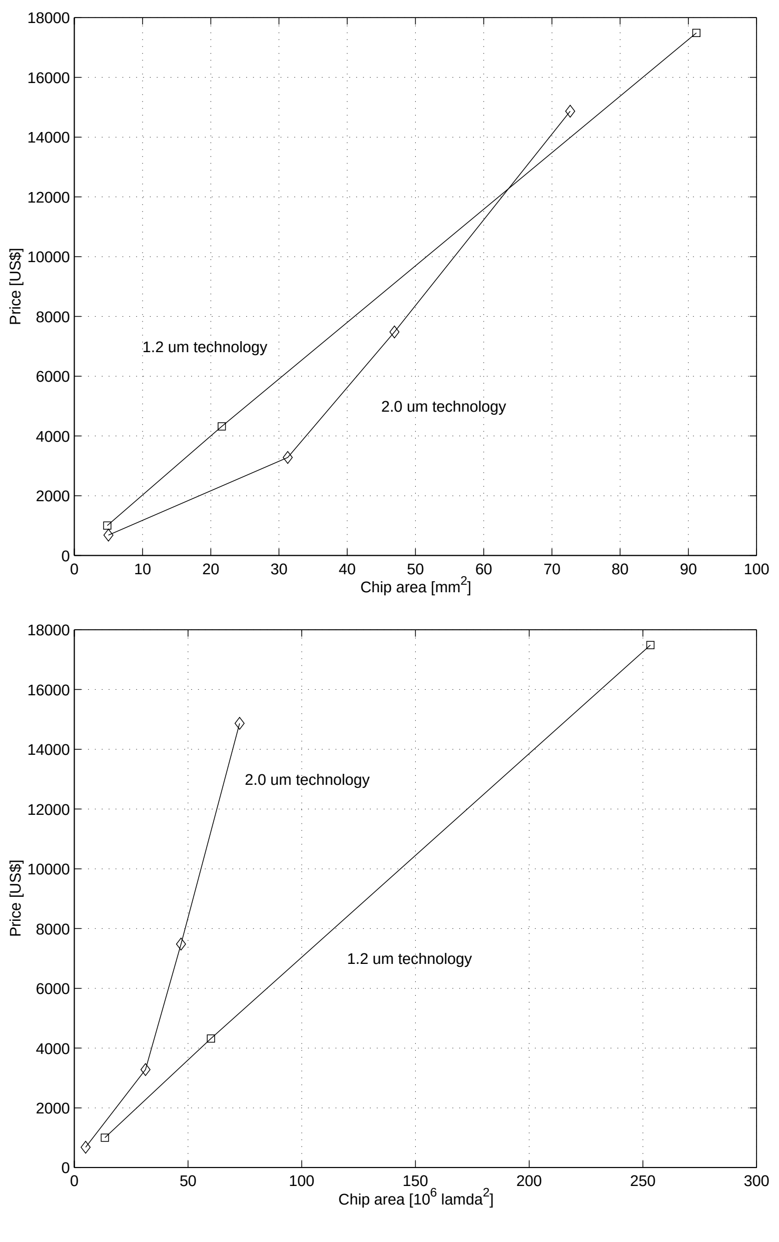


Figure 2.7: Chip price per area (top) and per lambda (bottom) for the 2.0μm and the 1.2μm technologies. Even if only chip area is considered, the 1.2μm process is cheaper for large chip sizes. More relevant is the comparison of the two technologies based on the lambda measure. In this case for any chip size the 1.2μm process is cheaper.

2.5 Previous Work on Motion Sensors

In analog VLSI quite a few sensors have been built for pure spatial image processing, some of which show a fair degree of robustness. Examples here are the retina by Delbrück [DM94] with adaptive photoreceptors, the retina by Mahowald [Mea89] with center surround response, the retina by Boahen [Boa96a] with center surround response, fuse networks such as in [HKL90] for detecting image discontinuities and centroid computation schemes such as in [DeW92]. For spatio-temporal image processing only very recently analog VLSI sensors have been reported which perform satisfactory in real world environments. Part of the reason is that in almost all motion detection schemes some form of delay or storage element is required, which apart from being area consuming, are difficult to implement. In this section some earlier implementations of motion detection schemes are discussed.

Lyon's eye

Richard Lyons optical mouse [LH82] was one of the earliest smart vision sensors designed and implemented on silicon. Lyon's 'eye' is basically a digital motion detection chip. Photocurrent over a photodiode is integrated over time on a capacitor, and transduced to a digital signal by a simple inverter. The time for charging up the capacitor depends on the input light level. Through lateral inhibition a cell exposed to a brighter pattern which reaches a 'high' state earlier will suppress the neighbors. Thus by tracking the location of the winner cells the movement of the input image, which in this case was the very defined fixed pattern on the mouse pad, can be determined.

This implementation was visionary in the sense that through the clever choice of the lateral inhibition the imager became almost independent of the light level and the reset clock was generated through a self-timed handshake with the imager array. Lyon's 'eye' showed that a motion detection implementation in analog VLSI for a given very specific environment was possible. The implementations discussed next were designed to perform with more arbitrary stimuli, which represented a much harder challenge.

Tanner and Mead's optical flow chip

Tanner and Mead [TM86] [Mea89] designed a chip which was supposed to solve the optical flow equation

$$\frac{d}{dt}I(x,t) = 0 \quad \Rightarrow \quad \frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0 \tag{2.3}$$

globally for the whole chip. Every pixel on the chip performs a computation to check whether the global velocity satisfies its local constraint given by equation 2.3. Depending on the error, every pixel supplies a force

$$\vec{D} = -\left(\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t}\right)\nabla I \tag{2.4}$$

that tends to move the global velocity estimate towards satisfying the local constraint in the direction $\nabla I/|\nabla I|$. The global velocity vector components are analog voltages on two global wires which are charged or discharged by the correcting forces represented as currents. As can be seen, the force D contains the error that a pixel experiences for the global velocity signal given its local temporal and spatial derivatives. If the error is large, then the pixel will pull the global velocity signals stronger in the direction of its local constraints. Interestingly for the force D this error signal is multiplied with the spatial gradient of the light intensity. This means that a pixel locally experiencing a high contrast environment will be pulling the global velocity signal stronger in its direction than a pixel experiencing a weaker contrast. Here the local contrast is taken as confidence measure. The underlying assumption is that in high contrast areas a velocity signal can be computed more reliably than in low contrast areas. With this approach the division by zero problem which is intrinsic to the gradient method can be avoided, because pixels experiencing low contrasts are prevented from influencing the global velocity signal. This approach cannot be used, though, in implementations of the gradient method where a local velocity information is desired. In this case other methods have to be found. For the CMotion1d sensor implementation, which will be presented in Chapter 3, solutions to the division by zero problem for a local velocity computation with the gradient method were found.

To the knowledge of the author the sensor implementation did non work satisfactory. This complex voltage mode implementation had problems due to device mismatch. Also for obtaining a local flow field the solution of a global constraint is not suitable. On the other hand if the local flow field is computed, as for example in the CMotion1d and Gradient2d sensors of this work, by averaging over the array an estimate of the global motion can be obtained, too.

Moore and Koch’s motion detector

Moore and Koch’s motion detector [MK91] was a variation of the Tanner Mead sensor. Through opening a feedback loop in the original circuit, now the temporal derivative was multiplied instead of divided by the spatial derivative. The algorithm was implemented in voltage mode in a 1-D array.

This chip compares to the Gradient2d implementation presented in chapter 4, which multiplies the temporal derivative with the sign of the spatial derivative in both spatial dimensions.

Delbrück’s velocity tuned motion sensor

A very different motion detection implementation is presented by Delbrück [Del93]. Unlike the schemes which depend on spatial and temporal derivatives, temporal correlation is used to extract motion information. The motion architecture extends the pairwise correlation model of a simple Reichardt detector by using a unidirectional delay line as a tuned filter for a particular velocity. The input to the delay lines is capacitively coupled in from photoreceptors. An edge passing over the delay line creates a traveling signal that spreads and decays with time. If the velocity of the edge is matched to the delay, the successive inputs to the delay line pile up in synchrony. Velocity tuning is obtained by taking the square of the local delay line output. In a two dimensional implementation a hexagonal architecture that encodes any possible direction-of-motion using three non negative values is used.

This chip was the first implementation of a functional 2-D motion detector which computes local motion information. The sensor could be very interesting for biologically inspired systems, the fact that it is tuned to a specific velocity, though, makes it less straightforwardly useful for machine vision like applications.

Horiuchi's delay line motion detection chip

The 1-D velocity detection chip reported by Horiuchi et al.[HBB$^+$92] is based on delay lines, too. The signal from each photoreceptor is passed through a temporal edge detector which generates a spike in response to a fast rising intensity. The outputs of each two firing neurons are passed through two delay lines with opposite orientation. A correlating circuit determines the location at which the two spikes meet. For very fast velocities the signals meet at the center of the delay lines, for slower velocities they cross either left or right of the center depending on the direction-of-motion.

Although this method seems promising for a multi-velocity sensor, as compared to Delbrück's delay line chip, only a 1-D velocity measure is reported. The algorithm and associated implementation make an extension to a 2-D hardly possible.

Kramer et al.'s FS velocity sensor

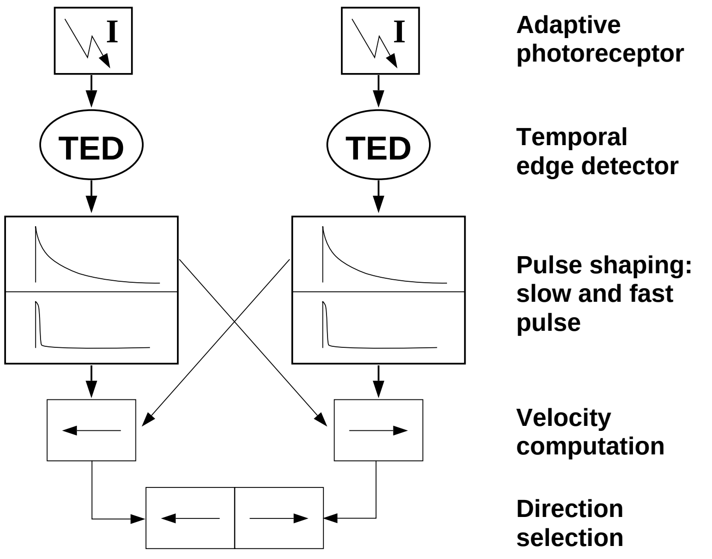


Figure 2.8: Block diagram of the elementary velocity computing unit of the Facilitate and Sample sensor (FS). The temporal edge detector (TED) generates a spike in response to a fast enough change in the light intensity. A slowly decaying facilitation signal and a fast and short pulse are generated in the pulse shaping stage. In the velocity computing stage with the fast pulse the slow signal of the neighbor is sampled. In the direction selection stage this sampled voltage is compared with the voltage of the other direction. Whichever is larger is reported as output.

Kramer at al.'s 1-D velocity sensor [KSK97] detects temporal edges and essentially computes a measure for the time of travel of an edge from one pixel to the next. A block diagram of the elementary velocity detector is shown in figure 2.8. At the time of the occurence of an edge at one pixel a capacitor is charged and starts to decay logarithmically. When the edge reaches the next pixel, the voltage of this capacitor is sampled. The slower the velocity of the edge was, the further the capacitor has discharged and the smaller the reported voltage is.

This sensor is one of the most successful implementations for 1-D velocity computation. Due to the compressive nature of the stimulus velocity to output function several orders of magnitude of velocity can be represented. The sensor performance is mainly determined by the performance of the temporal edge detector. The fast pulse length determines the fastest detectable velocity, the TED sensitivity determines the lowest detectable contrast and velocity. The sensor operates reliably even with natural stimuli if several velocity computing units are averaged. The reliability of a single velocity computing unit is not always satisfactory. In [BDK97] experiments are reported where the FS sensor was used in a natural environment to determine ego-motion, and in [BD97] the performance of the FS sensor to compute fast rotational velocity in a natural environment is investigated.

The FS sensor is not immediately suitable for a 2-D implementation for two reasons. First the unit pixel is large, mainly due to the slow pulse capacitor, and second the orientation tuning curve for two orthogonally oriented FS cells has discontinuities at 0^o, 90^o, 180^o and 270^o. This is because a horizontally oriented edge moving up or downwards causes a maximal velocity output in the x direction and similarly a vertically oriented edge moving right or leftward causes maximal velocity output in y direction.

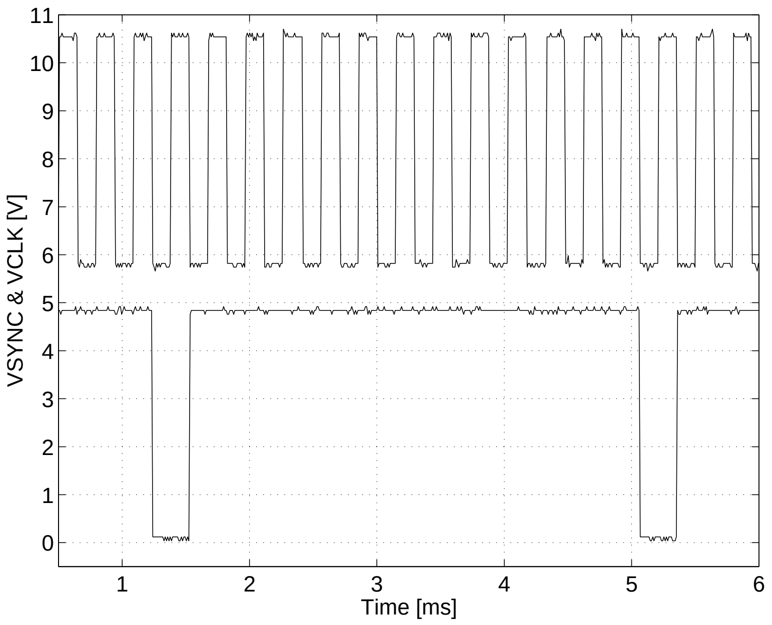
Kramer's FTI velocity sensor

Kramer [Kra96] reports a three pixel interaction scheme and 1-D implementation which reports a pulse-length coded time-of-travel signal. Like the FS sensor pulses are generated with a temporal edge detector in response to fast enough intensity changes. The local output goes high, indicating an edge is moving, when the edge passed two pixels sequentially (the facilitation and the trigger pixel). The passage of the edge over the third pixel (inhibition pixel) will reset the output.

This sensor is supposed to be more robust against false output because three subsequent pixels have to interact before an output is generated. On the other hand the reported velocity field becomes sparser. If a value coded time-of-travel output is desired, an integrator has to be added, thus increasing the size of the pixel. Although a 2-D implementation of the FTI sensor has been reported [HK97b], the FTI scheme is not very suitable for high density arrays because due to the three pixel interaction scheme a large part of the layout is used by just wires.

Etienne-Cummings et al.'s motion detection chip

Etienne-Cummings' velocity chip uses a modified Reichardt algorithm [ECdSM97]. In this algorithm, called temporal domain optical flow measurement, the image intensity zero-crossings are detected and their appearance and disappearance is determined. Velocity is computed by timing the interval between the disappearance at one pixel and the reappearance at the next pixel. The algorithm uses only nearest neighbor interaction and requires only the sign of the spatio-temporal derivatives and 1 bit Boolean multiplications.



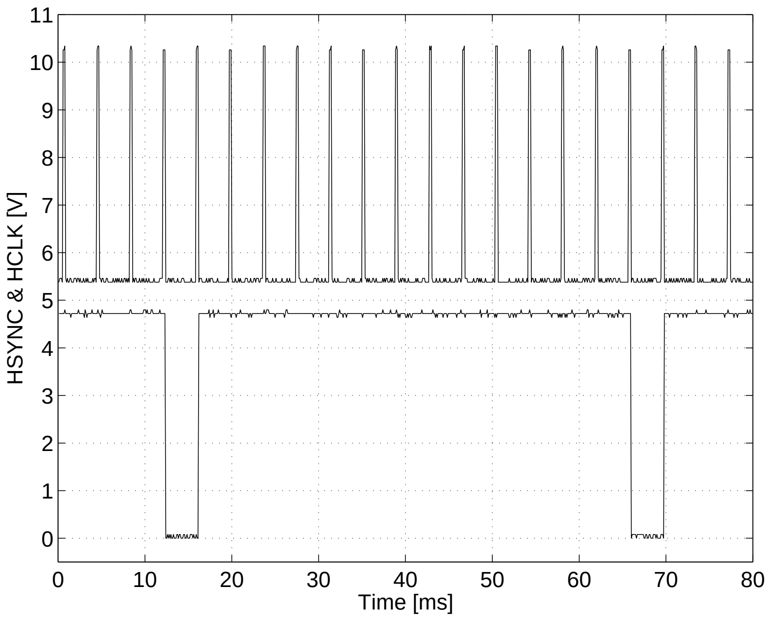


Figure 2.9: Scanner signals for slow scanning speed.

2.6 Features Common to All Sensors

2.6.1 Scanners and real time averaging

The 1-D and 2-D sensors presented in this work compute the local light intensity and motion signal continuously in time. This means that at every instance in time every pixel can be read out, the pixels do not need to be clocked, and the readout is non destructive. On the other hand because of

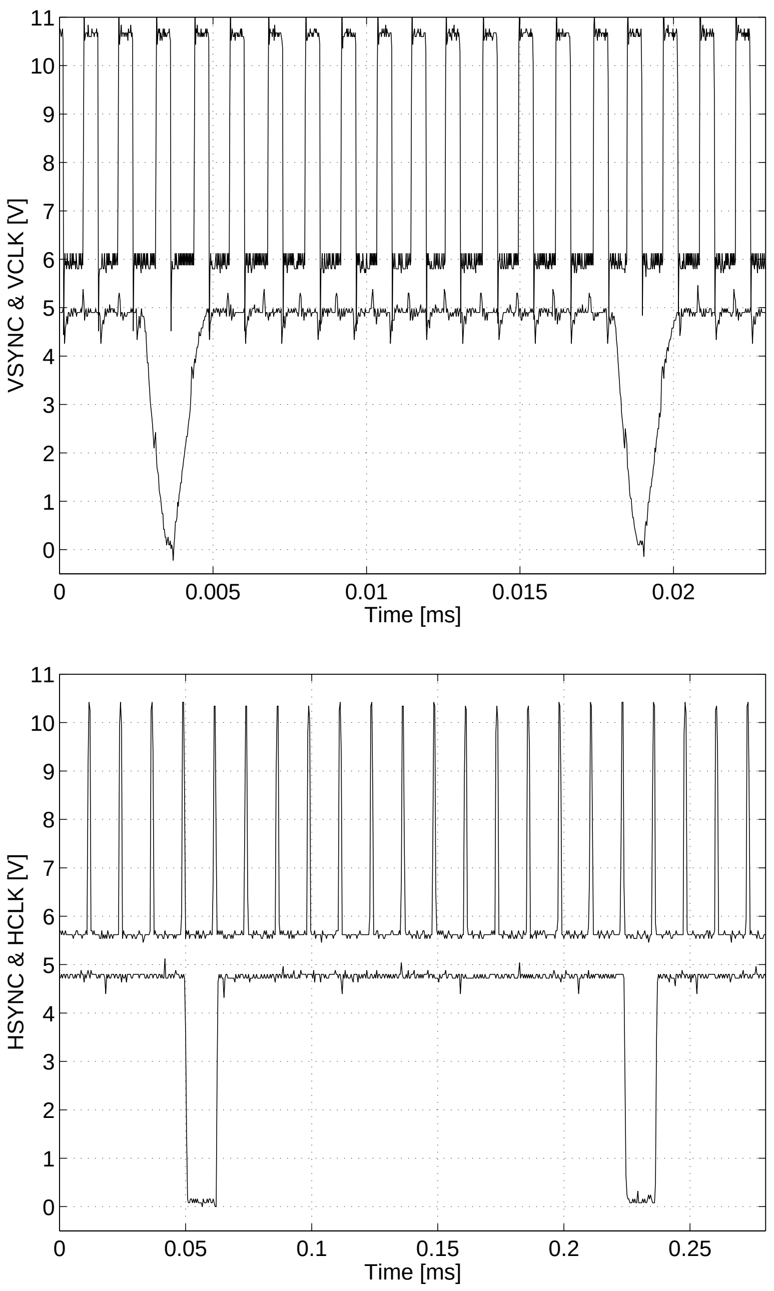


Figure 2.10: Scanner signals for fast scanning speed.

the high number of pixels not every pixel can have its own output pin on the chip. Serial scanners [MD91a] [MD91b] are used to probe the outputs pixel by pixel, and a clock is used to step from pixel to pixel. One serial scanner is an alignment of shift registers. At every clock cycle the bit is advanced by one location. For the vertical scanners on a 2-D chip the active register turns on its corresponding row of pixels. The horizontal scanner selects one row, and thus one of these pixels can be read off the chip. For a 1-D chip only the horizontal scanner is required. If the bit in the shift register falls out of the last register, the SYNC signal goes low, and with the BITIN signal a new bit can be loaded into the first register.

These scanners can conveniently be used for averaging over any rectangular patch of pixels on a 2-D array. Instead of inserting with the BITIN signal just one bit into the shift registers, by inserting several bits into the vertical and horizontal scanners, several pixels are added together on the output line. That way real time averages are achieved.

With the FTC sensor the vertical clock speed could be driven as high as 1 MHz with the scanner still operating correctly. For a sensor of 12×13 pixel resolution this clock speed would result in a frame rate of 5.5 kHz. This frame rate is much higher than achieved by any other motion detection system. The sensors presented in this work, though, could not be tested with this high frame rates. The reason is, that the analog to digital converter and the computer necessary for displaying or storing the data were only able to be operated down to scan times of minimal 200μs, which resulted in a maximal frame rate for the FTC sensor of 30 Hz. Additionally for high frame rates the digital noise might confound the performance of the analog circuitry in the sensor pixels.

In Figure 2.9 scanner signals for a commonly used clock speed of 3.4 kHz are displayed. The clock is generated by a function generator. All signals have little transients and are truely digital. For a scanning speed of about 1 MHz (see Figure 2.10) the digital noise is much stronger and the vertical SYNC pulse has strong transients. For a faster scanning speed than 1 MHz the vertical SYNC cannot follow any more.

2.6.2 The question of persistence time

A question which is not commonly discussed is how the output of the motion sensor should be represented in time. This issue is important for any further use of the motion signal, and it is related to how the signals are reported off the sensor. Most of the time, though, the output representation in time is governed by the particular algorithm of the sensor. Six possible schemes can be classified and are actually used in hardware implementations:

1. **Transient motion output**. The motion sensor output is reported only during the time when a motion signal can actually be detected in the visual scene. This is the case for most if not all gradient based methods and also the Reichardt type correlation based methods, if the temporal low pass filter time constant at the output is not too long. The Gradient2d sensor reported in this work in chapter 4 is an example which has this kind of output. Since here the temporal derivate is multiplied by the spatial derivative, there will only an output be generated if a temporal change is visible. Of course there can be temporal change due to flicker noise, which also causes motion output. But the advantage of this transient output is, that in general the local motion output will only be non zero if there is actual movement. If the local motion stops, instantaneously the motion output will be zero, too. The drawback, though, is that motion information might be missed if the sensor output is serially scanned of the sensor. Imagine a sharp edge moving quickly over a gradient based senor. The motion information will only be computed in the immediate neighborhood of the edge, where temporal and spatial

derivatives are non zero. If the sampling is too slow, only locations with zero motion might be probed and the edge might not be seen at all.

2. **Transient velocity output**. For the implementation of the gradient method, as presented for the CMotion1d sensor in chapter 3, not only the spatio-temporal motion is reported correctly, but even the spatio-temporal velocity. This means that at any given instant in time the sensor can be probed at any place and the obtained output will represent the local velocity at that same time. If the output is zero, then either the scene is locally stationary or there is movement which cannot be detected due to missing contrast. But again, as discussed for the transient motion output, velocity information can be lost to outside of the sensor if it is not read in time.

3. **Transient pulse width coded velocity output**. This scheme is naturally used by some feature based sensors [Kra96] [SBK93]. For example the pixel output signal can assume an activated state during the time of travel of an edge between two pixels, and be zero otherwise. Although mathematically this pulse length would only to be integrated over time, in practice, when a scanner is used to probe every pixel for its state, a pulse length output is less useful. The reason is that the scan rate then determines the accuracy with which the beginning and the end of the pulse can be detected. Again, for fast moving objects their velocity signal could be missed entirely to the outside of the chip. For pulse width coding sensors a common solution is to add integrators to every pixel [ECdSM97]. This increases the pixel size and lessens the chance for high density 2-D arrays.

4. **Sample and hold velocity output**. Some schemes take the approach to hold the once computed velocity output until new velocity information is available. This is the case for some feature based velocity sensors as in [KSK97]. The obvious advantage of this approach is that a velocity signal that has been detected on the sensor is not lost, however late it is read off the chip. The serious drawback, though, is that once a particular velocity has been computed and stays, the observer cannot know if velocity information is constantly being refreshed or the moving object is way out of sight already. In practice an even more serious problem appears, because the held output will decay slowly over time. This means for a once computed velocity the observer sees an apparently slower and slower velocity output. The sample and hold scheme can be successful if it is granted that a high contrast environment is explored and there is constantly motion either due to objects in the environment or due to ego-motion.

5. **Finite signal length motion output**. For one of the sensors presented in this work (the FTC sensor, see chapter 5) a new scheme for the output has been invented. This was done to achieve both the advantages of the transient and the holding scheme. The output is neither fully transient nor held over a long time, but a finite persistence time can be programmed. The persistence time can be set such that with the given scanning speed no events are missed, but on the other hand after motion ceases the motion output goes to zero in a reasonable time. The longer the persistence time is set, the more of the leading and trailing edge signal of an object will remain on the sensor output for any given object velocity. This shows the disadvantage of a fixed persistence time: The sensor output will be very sparse for a slowly moving object,

because few new motion signals are generated per time. A fast moving object, on the other hand, will cause the sensor to be 'striped' with the objects' motion information.

6. **Adaptive signal length motion output**. An adaptive persistence time can ameliorate the afore mentioned problems. For a velocity sensor the velocity output should be held longer for smaller speeds and shorter for faster speeds. Effectively the length of the trail of velocity vectors behind an object should be independent of the objects' speed. This scheme has not yet been implemented.

2.6.3 Chip testing setup

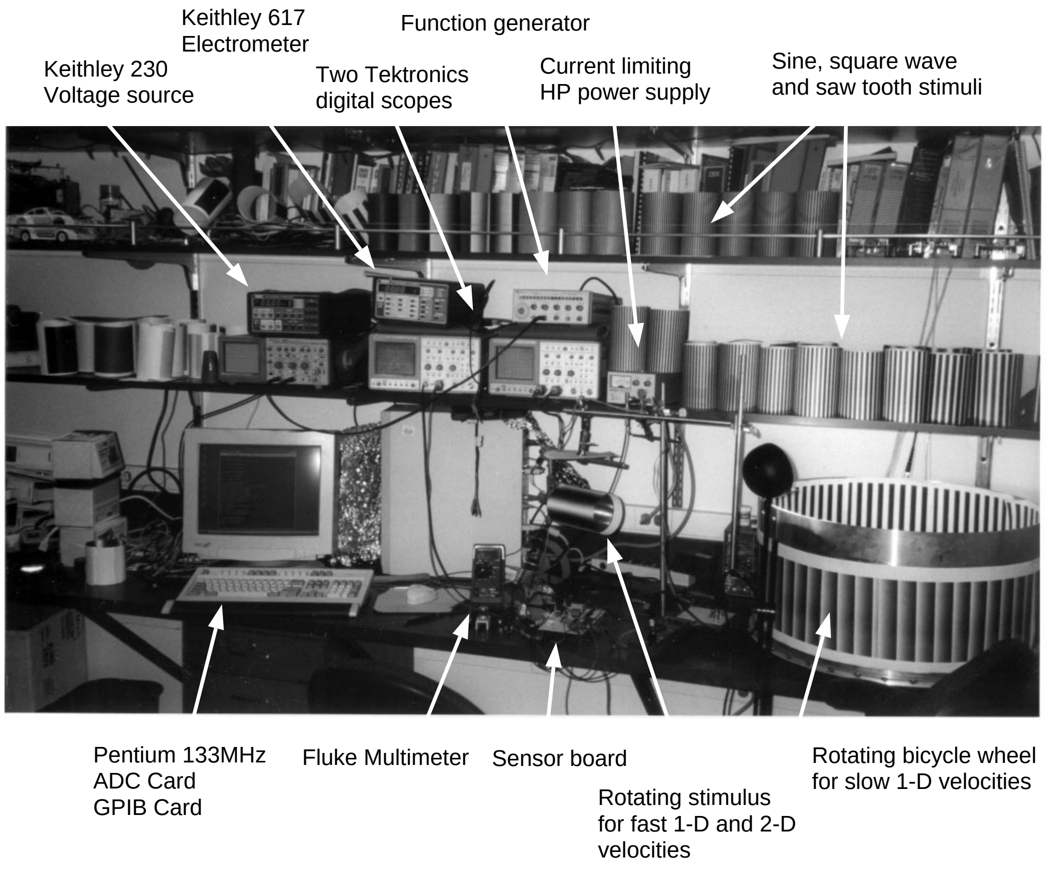


Figure 2.11: Chip testing setup.

The following equipment was used to test the sensors presented in this work (see also the photograph of the lab bench in Figure 2.11):

- Pentium PC with 133 MHz Intel inside. The computer was used for data acquisition from the scopes via the GPIB card and from the sensor boards directly, via an analog-to-digital card. The computer was also used to drive the digital scanner circuitry on the sensor boards and display the velocity fields on the screen in real time. For real time display of velocity vector fields the computer represented the time bottleneck. A faster computer and faster video card would be necessary for displaying flow fields of higher resolution.

- Analog to digital and digital to analog card ADIO 1600 (Industrial Computer Source, San Diego, CA). 8 differential analog channels, resolution 12 bits by successive approximation, software selectable voltage ranges, conversion time 8μs, additionally 3$\times$ digital in/out ports. The outputs of the sensor chips are small currents, which are amplified and transformed into a voltage signal with current sense amplifiers. These voltages were read with the AD card into the computer for displaying or storage. Special purpose software was written in C for communication with the AD card.

- High speed GPIB card INST 2000 (Cyber Research, Burlington, MA). Used for communication with the digital scopes and the Keithleys. Special purpose software was written for communicating with these devices and acquire and save scope traces.

- Keithley 230 programmable voltage source.

- Keithley 617 programmable electrometer. This electrometer is able to read currents as small as a few tens of pico amperes, thus even the small currents generated in subthreshold operation of MOSFETs can be detected.

- Two Tektronics 2430A digital 2 channel oscilloscopes. Often times several signals had to be displayed at the same time. A four channel scope would be preferable, such that 4 traces can be grabbed at the exact same time. Digital scopes allow to conveniently halt a trace, compute voltage levels and time differences on the screen. With special purpose software scope traces were acquired and transferred to a SUN Sparc station, where the scope traces were analyzed with Matlab programs.

- HP3312A Function Generator. For some 2-D sensor chips the function generator was used to generate the clock signal. The clock can also be generated with the computer and the digital output of the DAC, but for high speeds there was noise introduced due to the fact that the computer could not provide a constant clocking frequency. Clocking frequencies up to 6 kHz were used.

- HP 6214C 0-10V 0-1A current limiting power supply. The current limiting feature is important for not overloading the chips when they latch up, when they try to sink too much current, or when they are connected wrong.

- LPAC 5V and $\pm$12V regulated power supply. Can be used if the chip is well understood and the current limit is not required. The addition of some capacitors between Vdd and GND are advised to reduce noise. The $\pm$12V are convenient to drive operational amplifiers such as the TL074 which requires a bipolar voltage supply.

- ‘Can’ stimulator. Used extensively to generate commonly used velocities, capable of generating 1 and 1/2 orders of magnitude of velocities.

- ‘Bicycle wheel’ stimulator. Used to generate very slow velocities.
- Lenses 2.6mm-1:1.6, 4mm-1:1.2, 8mm-1:1.2 (Computar) and 13mm-1:1.9 (Zeiss) on special lens mounts to fit on 40 pin DIP ZIF chip sockets. Used to focus the incoming light onto the sensor chip.

Stimulus presentation devices

In order to properly characterize 1-D and 2-D motion sensors it was necessary to generate a controlled visual input. A stimulus device was built which was able to present arbitrary periodic stimuli oriented in an arbitrary direction with stimulus speeds covering almost one and 1/2 orders of magnitude. A aluminum can was attached to a DC motor and covered with the desired stimulus printed on paper. The motor was attached to a stand such that it could be turned in the horizontal plane. Just to give an impression of how large this velocity range already was: The stimulus device was able to generate velocities similar to what one could achieve by moving a hand as slow as possible and as fast as possible. The DC motor had a tachometer output and was connected to a variable voltage source. The tachometer output was rectified and low pass filtered such that a DC voltage as a function of motor speed was obtained. The motor was calibrated and the obtained fit function was used for the velocity experiments in this work. Figure 2.12 shows the relation between stimulus speed and tachometer voltage. On chip speeds ranging from 2mm/sec up to 78mm/sec could be achieved.

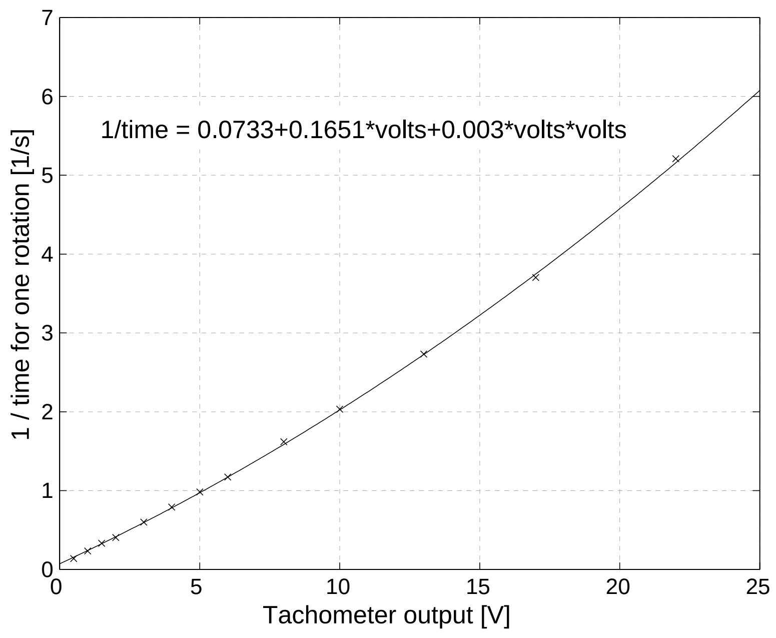


Figure 2.12: Stimulus velocity calibration. The velocity of the ‘can’ stimulus was calibrated by measuring the time for one turn and the reported rectified and low pass filtered tachometer output.

In order to test the motion sensors with even lower stimulus speeds, a similar DC motor was attached to a horizontally rotating bicycle wheel. On the wheel an aluminum rack was mounted onto which the paper stimuli could be attached. A mechanical gearing was used to further slow down the velocity. On chip speeds ranging from 0.05 mm/sec up to 2 mm/sec could be achieved. These speeds were barely if at all visible with the eye.

2.6.4 Definition of contrast

For all experiments where contrast measures are given, the following definition has been used:

$$C = \frac{I_{max} - I_{min}}{I_{max} + I_{min}} \tag{2.5}$$

where C is the contrast, I_{max} and I_{min} are the locale intensity maximum and minimum, respectively.

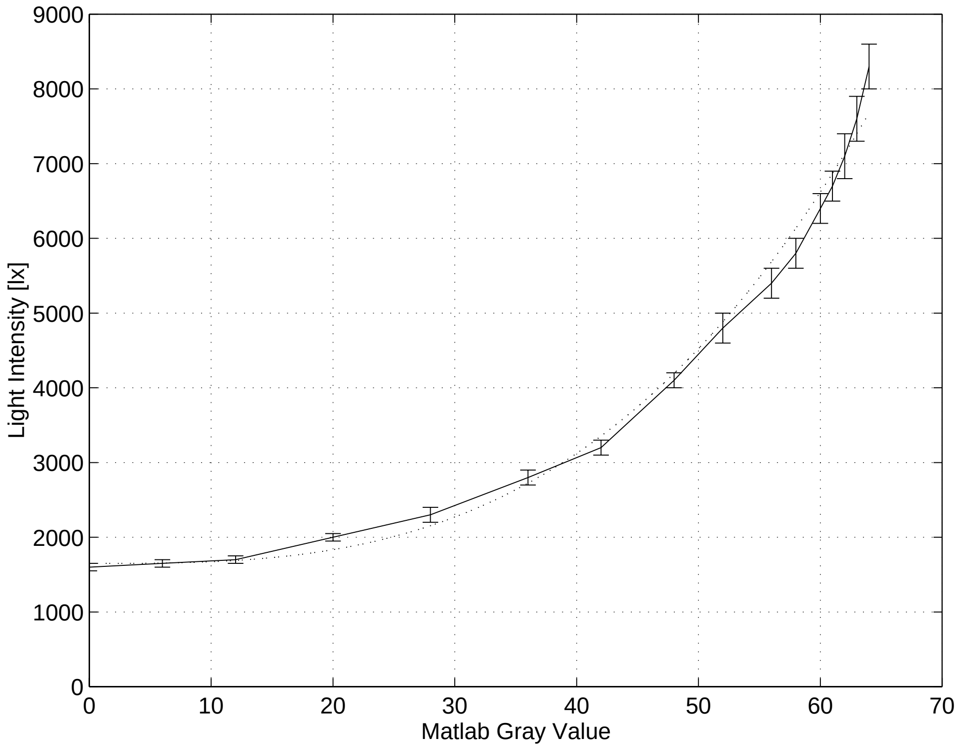


Figure 2.13: Light intensity and contrast calibration. In Matlab with different gray values full field stimuli were generated and their intensity measured. The obtained cubic fit function was used to compute contrasts for arbitrary pairs of Matlab gray values using the definition in equation 2.5.

The intensity curve was obtained using a light meter (Ultra-Pro, Gossen, Germany). In Figure 2.13 the measured light intensity is plotted over the gray values used in Matlab to generate the stimuli. The printer was a Lexmark Optra Lxi+ with new toner cartridge, resolution 600dpi. The fit function was found to be:

$$I = 1650 + 0.023\, g^3 \tag{2.6}$$

where I is the fitted light intensity and g the gray value, as used by Matlab. $g = 0$ is black and $g = 64$ is white. The calibration was performed for different ambient light levels and yielded very similar fit parameters.

Temporal contrast

For vision sensor implementations the definition of a temporal contrast C_t is useful

$$C_t := \frac{\partial I}{\partial t}. \tag{2.7}$$

where I is the light intensity integrated over the photoreceptor area. Every vision sensor implementation uses a lens to focus the image on the photoreceptor array, and every photoreceptor necessarily has a finite size. For these reasons every edge if moved over one photoreceptor causes a smoothly varying intensity pattern over time. Moving the edge faster is equivalent to moving an edge with higher contrast but the same speed, where equivalence means $C_t = const$. The definition of the temporal contrast will be particularly useful for the discussion of the temporal edge detector in chapter 5 on the FTC sensor.

2.6.5 Ambient illumination

Although for the human eye barely noticeable after adaptation, the lighting level in an artificially lighted room can differ from outdoors on a sunny day by many orders of magnitude. For the vision sensors in this work an adaptive photoreceptor is used, which is able to adapt to some extent to the ambient light level. Nevertheless differences in sensor operation were observed when they were operated under vastly different lighting conditions. Therefore the average illumination level for the laboratory was measured in which the sensors were tested. For most of the experiments the stimulus brightnesses were between 50 lx to 250 lx.

All experiments have been performed in artificial lighting, which was supplied by light bulbs. A normal reading halogen lamp was used for increased background illumination. It is known, that some sensors perform more reliably under DC lighting. This is particularly true for sensors that rely heavily on any sort of temporal differentiation. No systematic study of the influence of AC lighting has been performed in this work.

Chapter 3

CMotion1d Sensor: Gradient Method for 1-D Velocity

3.1 Introduction

In this chapter a velocity sensor is presented which is an implementation of the gradient method for velocity computation in 1-D. To the author's knowledge this is the first working sensor of its kind. The velocity is computed locally by the division of the temporal and the spatial derivative

$$\boxed{v(x,t) = -\frac{\partial I/\partial t}{\partial I/\partial x}} \tag{3.1}$$

This formula follows readily from the local constancy assumption of a moving image brightness pattern $I(x,t)$

$$\frac{dI(x,t)}{dt} = 0 \tag{3.2}$$

where by using the chain rule for differentiation it is found

$$\frac{\partial I}{\partial x} v + \frac{\partial I}{\partial t} = 0 \tag{3.3}$$

from where equation 3.1 follows immediately.

Every three adjacent pixels form one velocity computing unit (see figure 3.1). The temporal derivative of the light intensity is computed on the central pixel. The difference of the light intensity of the two neighbors to the left and right is used as discrete approximation of the spatial derivative of the light intensity. The division circuit receives both inputs and computes the velocity signal in real time. Even though three pixels form one unit, only two communication wires run between every pair of pixels, and only nearest neighbor interaction is required.

The pixels are arranged in a circle on the chip such that the local velocity in one dimension can be measured as well as the averaged global rotational velocity.

This chapter is divided up in the following way: In section 3.2 an outline will be given on where the difficulties are for implementing the gradient method in analog VLSI and how they can

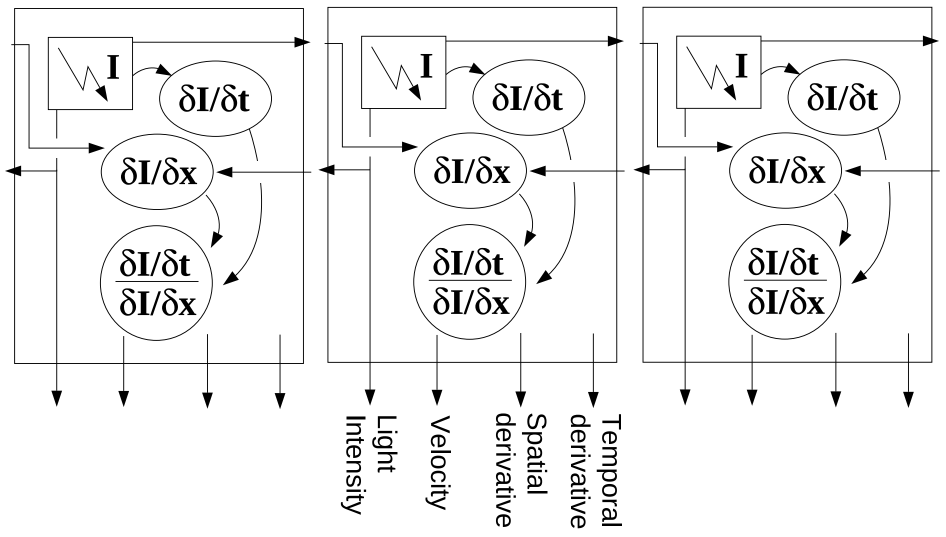


Figure 3.1: Block diagram of the CMotion1d sensor. Main components are the adaptive photoreceptor, the temporal and spatial derivative circuits and the division circuit. Only two signals are passed between each two pixels: The photoreceptor voltage to and from the neighbor. From every pixel the light intensity, both derivatives and the velocity signal are obtained.

be overcome. In section 3.3 the circuits used in the implementation will be discussed. The layout of both the pixel and the chip will be shown. In section 3.4 fundamental limits of a hardware implementation of the gradient method will be discussed. In section 3.5 the sensor operation will be illustrated with sample outputs. Important parameters will be treated. In the following sections experimental results from single velocity computing units will be presented. The velocity output dependence on the stimulus velocity, on spatial frequency, on contrast and on stimulus orientation will be demonstrated. In section 3.11 the response of the sensor for rotational velocity will be discussed. Here the output of all velocity computing units will be used. In section 3.12 an extension of the 1-D gradient method to a 2-D implementation will be outlined. The chapter will close with a summary and discussion.

3.2 Challenges

There are several non trivial problems to be solved for a successful implementation of the gradient method.

1. *Photoreceptor* The front end photosensitive device which transduces the incident light into an electrical signal sets the limits for all subsequent computation. It sets limits on the speed, noise and sensitivity of the entire sensor. Therefore it should have a desirable filter behavior, a large signal to noise ratio and should ideally be adaptive to the DC level of the incident light.

2. *Temporal derivative* Time domain computation as compared to space domain computation is difficult in hardware because storage elements are required. For the gradient method a circuit is needed which computes an *accurate* temporal derivative.

3. *AC flicker* A way must be found to avoid the temporal derivative circuit from responding to the AC flicker of artificial light as it is present in indoor applications.

4. *Spatial derivative* An *accurate* spatial derivative must be computed. Earlier attempts to implement the gradient method [TM86] showed that device mismatch poses a particularly hard problem for derivatives. For example a small linear range of the spatial derivative circuit together with a sufficiently large photoreceptor offset can yield a wrong spatial derivative.

5. *Division* Given the temporal and spatial derivatives, a circuit is required which computes the division of those two values accurately and continuously. This has been one of the most challenging parts in the past. It has to be realized, that the velocity output is computed in continuous time, unlike machine vision methods where time is discretized, and also unlike the feature based velocity sensors where the motion information is computed once and then held for a fixed time.

6. *Sign of velocity* If the computation in hardware cannot be performed with signed values, rectified values are used. Then a way must be found to achieve the correct sign with the division operation.

7. *Division by zero* For the division operation the division by zero case has to be considered and treated.

8. *Timing* The timing of temporal and spatial derivatives has to be very precise, otherwise the result of the division will be wrong. This requires that the temporal filtering behavior of the spatial and temporal derivative computation branches are matched.

9. *Finite size effects* The effects of implementing an infinitesimally derived mathematical method on a finite substrate must be considered.

In brief the following solutions were found to cope with these challenges (in the same order):

1. Although a photo-diode or a lateral bipolar photo-transistor in source-follower configuration as a front end photosensitive device would require much less chip area, the active photoreceptor developed by Delbrück and Mead [DM94] was chosen for the present implementation. Several features make this receptor attractive, for example it adapts to the DC light level, its response remains fast even for dim lighting and the AC flicker of artificial lighting can be filtered out.

2. There exist several ways to implement a time derivative of a signal in VLSI hardware. In [Mea89] a circuit is proposed which uses only one wide range transconductance amplifier, two transistors and one capacitor. This circuit, though, was used in voltage mode. Horiuchi [Hor97] successfully used a variant of this circuit in current mode which only requires a five transistor transconductance amplifier together with the two transistors and the capacitor. Both signs of the temporal derivative can be obtained and amplified in current mirrors. This temporal derivative circuit is used in the present implementation.

3. For suppressing the AC flicker of artificial lighting the temporal low pass filter behavior of the adaptive photoreceptor was used. The bias on the receptor has to be set such that the cutoff frequency is low enough to damp the 60 Hz flicker. The bias on the temporal derivative circuit and the aperture of the lens also affect the responsiveness to AC flicker.

4. For the CMotion1d velocity sensor a novel wide linear range transconductance amplifier was developed. The circuit is based on the five transistor transconductance amplifier and uses floating gates for a capacitive divider on the input nodes. The spatial derivative is locally computed by taking the difference of the light intensities of the two adjacent pixels. The spatial derivative is linear with the input intensity difference. Because of the wide linear range of the spatial derivative circuit and because of the adaptive behavior of the photoreceptor spatial derivatives can be computed very reliably.

5. The wide range of input and output signals that is naturally required for a division operation made it a very difficult task for a voltage mode implementation in previous designs [TM86]. For the present sensor the division operation is implemented using a four transistor current mode νMOS translinear circuit which is based on recent developments at Caltech [MDHM96]. The division circuit computes the correct result over several orders of magnitude of input signals.

6. The current mode division circuit can only perform a one quadrant operation. In the sensor therefore the absolute value of the spatial derivative is computed and then the division operation is performed. For sign correction a circuit has been developed which consists of a high gain differential amplifier and two digital transistors. In the present version of the sensor only one sign of the temporal derivative is considered.

7. A division by zero occurs in the gradient method whenever there is no feature present in the visual scene such that the spatial derivative is zero. In order to cope with this problem, a constant value can be subtracted from the absolute value of the temporal derivative (the numerator) and another constant value can be added to the absolute value of the spatial derivative (the denominator). Both constants can be controlled arbitrarily. The subtraction is such that the signal cannot become negative.

8. Time constants associated with the temporal and spatial derivative circuit are induced by parasitic capacitances. This has to be considered in the layout of these circuits. As will be shown in section 3.6, temporal and spatial derivatives are computed correctly in real time such that the velocity output is constant as long as a feature moves over the sensor. The effect of temporal filtering is discussed in greater detail in section 3.4.

9. Due to the discrete implementation in VLSI hardware spatial aliasing can occur. In order to increase the spatial aliasing frequency the photoreceptors are spaced as closely as possible. Defocusing the lens can be used to reduce the effect of aliasing.

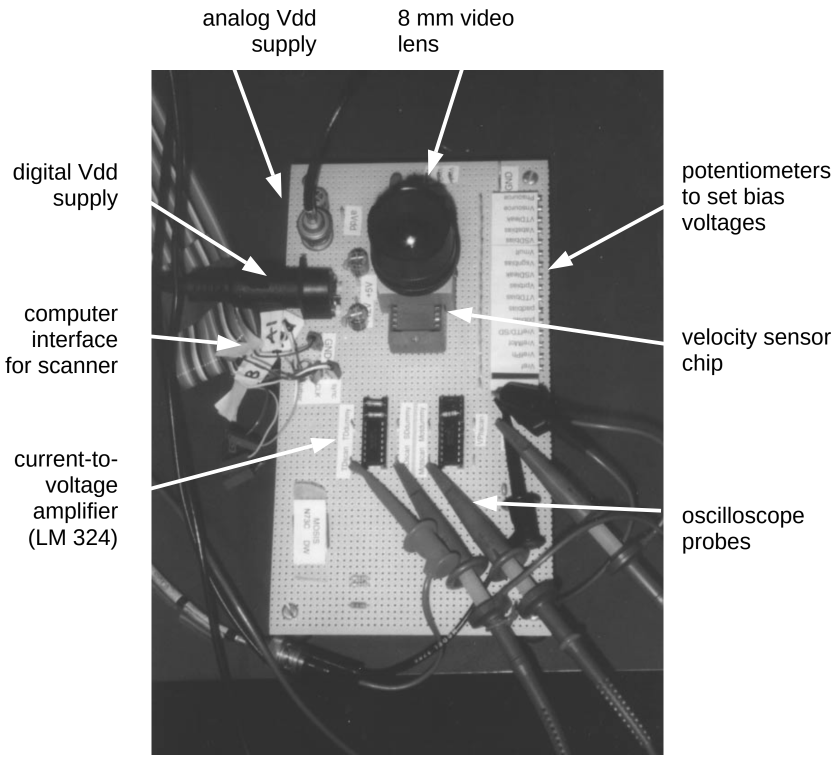

analog Vdd supply
8 mm video lens
digital Vdd supply
potentiometers to set bias voltages
computer interface for scanner
velocity sensor chip
current-to-voltage amplifier (LM 324)
oscilloscope probes

Figure 3.2: Photograph of the CMotion1d board.

3.3 Circuit and Layout

Figure 3.2 shows a photograph of the complete CMotion1d sensor system. The only additional circuitry needed besides the actual sensor chip are amplifiers for converting the chip output currents to voltages and circuitry or a computer to drive the scanners. It is not necessary but advisable for noise reasons to use separate power supplies for the digital and analog circuitry. The whole system can be powered from a 9V battery. The potentiometers are used to generate the bias voltages. Gate bias voltages are generated with 50 kΩ potentiometers, for current sources 5 kΩ are used to avoid voltage drop over the resistor. In the following the building blocks within the velocity computing pixels are described.

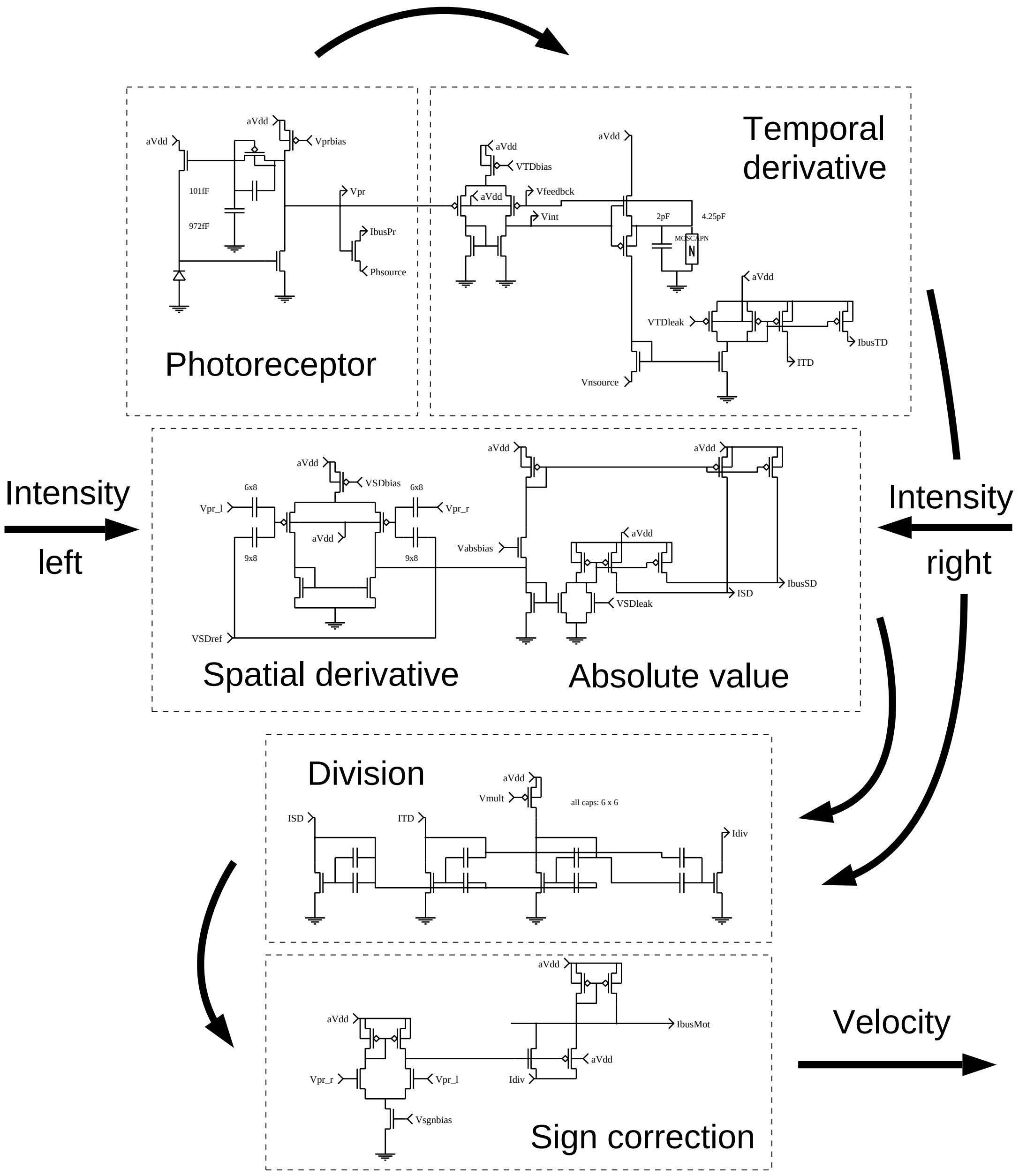


Figure 3.3: Pixel schematic of the CMotion1d sensor. The temporal derivative circuit receives input from the local photoreceptor, whereas the spatial derivative circuit and the sign correction circuit receive input from both neighbors. Temporal derivative and the absolute value of the spatial derivative are divided in the central νMOS circuit. The velocity signal as well as the derivatives and the light intensity are reported by every pixel.

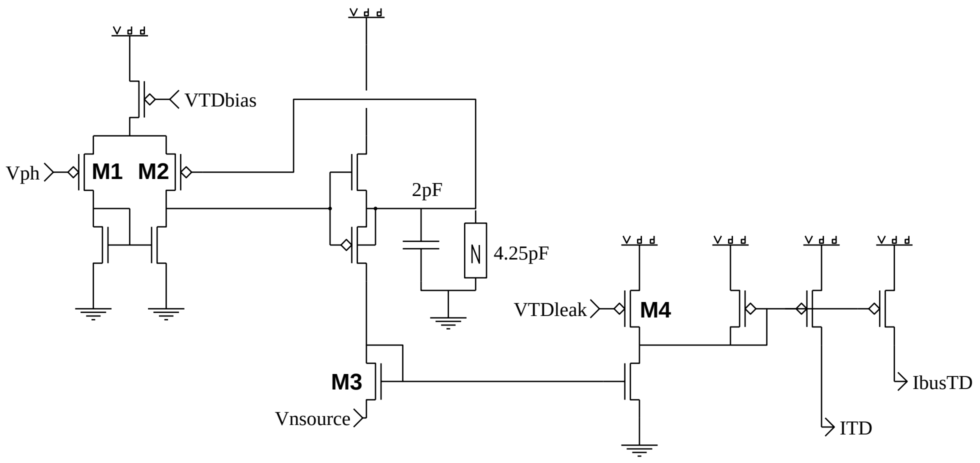


Figure 3.4: Temporal derivative circuit. Transistor M4 is used to subtract a constant current off the temporal derivative current. The capacitors have the total size of $6.25\,pF$.

3.3.1 Adaptive photoreceptor

For the front end photoreceptor the adaptive receptor developed by T. Delbrück [DM94] was chosen. The receptor was layed out very conservatively. Poly1-Poly2 capacitors of the sizes $972\,fF$ and $101\,fF$ were used, yielding a capacitive divider ratio of 9.6. The output of the receptor was not buffered, unlike in the Gradient2d sensor. Therefore some influence of parasitic capacitances on the receptor were expected. Parasitic capacitances are induced mainly trough gate connections and wires. Gate capacitances are an issue for the CMotion1d sensor, because the photoreceptor not only connects to one temporal and two spatial derivative circuits, but also to two sign correction circuits. Indeed the long low pass filter time constants and the larger mismatch that were found indicate some influence of parasitic capacitances.

3.3.2 Temporal derivative circuit

For the temporal derivative a variant of a circuit introduced in [Mea89] was used. Instead of voltage mode the circuit was altered to work in current mode. This has previously been shown to work satisfactory by Horiuchi [Hor97]. Only one sign of the temporal derivative was considered for the present implementation which is sufficient for velocity computation if both spatial derivative signs are taken into account. Transistor M4 in Figure 3.4 was added to allow for subtraction of a constant current off the temporal derivative current. This allows for compensation of leakage currents and is an effective way to deal with the division by zero problem. An unusually large capacitor consisting of a Poly1-Poly2 capacitor of $2\,pF$ and a MOScap of $4.25\,pF$ was used. This was to increase the current through M3 such that the amplification with $V_{nsource}$ could be kept low, which in turn prevents from amplifying leakage currents.

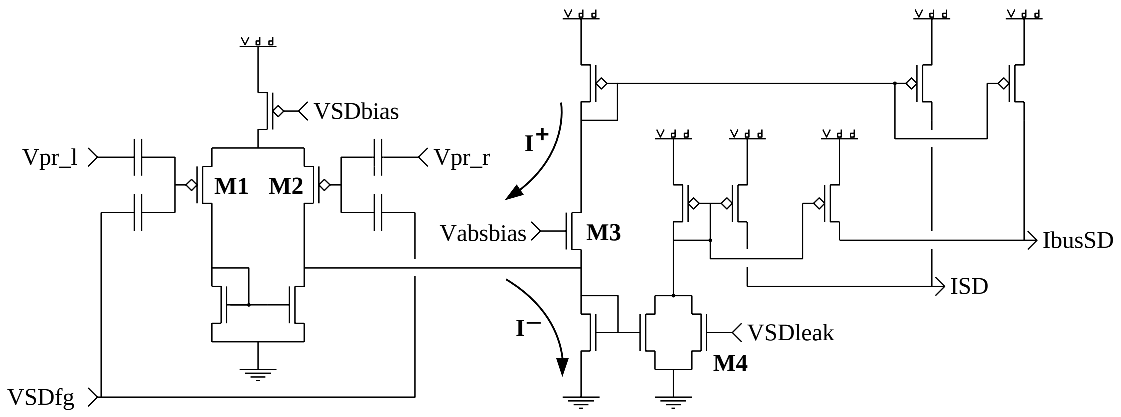


Figure 3.5: Spatial derivative and absolute value circuit. The two different current paths in the absolute value circuit are indicated: I^+ is flowing for $V_{pr\_l} < V_{pr\_r}$, the current flows through transistor M3 and is mirrored once. In the opposite case I^- is flowing and is mirrored twice. This asymmetry causes mismatch in the output of the absolute value circuit. The leakage transistor M4 is used for adding a constant current to the spatial derivative to cope with the division by zero problem.

3.3.3 Spatial derivative and absolute value circuit

The discrete spatial derivative I^i_Δ of the light intensity I at pixel i in the sensor is computed as

$$I^i_\Delta = const. \times \left(I^{i+1} - I^{i-1}\right), \tag{3.4}$$

therefore the discrete spatial derivative is correct up to the second order in Δ, with Δ the pixel distance. For computing the spatial derivative in analog VLSI a new wide linear range transconductance amplifier (transamp) was developed. This circuit will be described in detail in the next section. Functionally this amplifier receives two input voltages and generates an output current linear in their difference over a selectable wide range. The wide input range was obtained by using a floating gate capacitive divider. For the purpose of the CMotion1d sensor a divider ratio of $6 \times 8/9 \times 8 = 2/3$ was chosen to achieve a linear range over all possible input voltages from the photoreceptors. The output of the transamp is a bidirectional current, which has to be rectified for further computation in the νMOS division circuit. The rectification is performed by an absolute value circuit. For an analysis of this circuit using the translinear principle see [AB96]. Because in the absolute value circuit the bidirectional current coming from the spatial derivative circuit takes two different paths for the two possible signs, mismatch in the output is observed. This mismatch is caused by device mismatch in the current mirrors and the asymmetric transistor M3 (see figure 3.5). In effect the current generated by a given positive linear intensity gradient will be different from the current due to the same but negative intensity gradient. This in turn causes a gain difference in the velocity output as discussed in section 3.6. For example outputs of this phenomenon see also the sample output section 3.5. Transistor M4 is used for adding a constant current to the spatial derivative to cope with the division by zero problem.

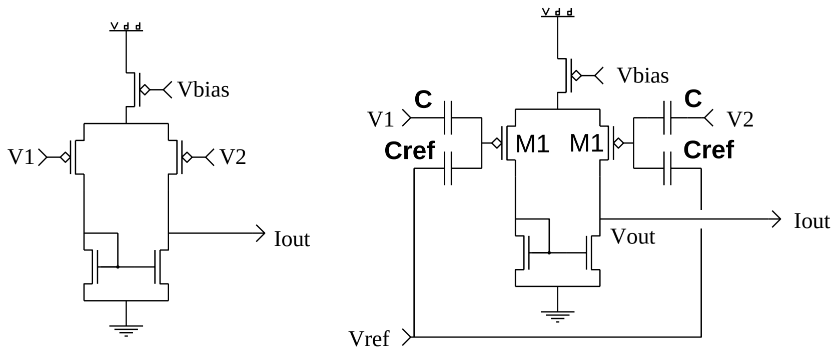


Figure 3.6: Standard (left) and wide linear range (right) transconductance amplifier. Due to the capacitive divider formed by C and C_{ref} the input voltages V_1 and V_2 in the wide linear range transamp are decreased and the linear range of output currents is increased.

3.3.4 Analysis of the Wide Linear Range Transconductance Amplifier

A new circuit was developed in this work which generates a current proportional to a voltage difference over an arbitrary wide range. The circuit is based on the transconductance amplifier (transamp) discussed in [Mea89]. The idea is to use capacitive dividers on the inputs of the transamp to lessen the influence of the input voltages on the differential pair.

The input-output characteristic of a transamp is

$$I_{out} = I_b \tanh \frac{\kappa\,(V_1 - V_2)}{2\,V_T} \tag{3.5}$$

where I_b is a constant, $\kappa \approx 0.7$ is a constant determining the effectiveness of the gate voltage on the channel, $V_T = k\,T/e$ the thermal voltage with k the Boltzmann constant, T the temperature and e the unit charge and V_1, V_2 are the two input voltages. This expression holds for sub-threshold source-drain currents

$$I_{sd} = I_o\, e^{\frac{\kappa\, V_{sg}}{V_T}} \tag{3.6}$$

where V_{sg} is the the source-gate voltage and the transistor is in saturation. The equivalent expression for inputs coupling onto a floating gate is derived in [MDHM96]:

$$I_{sd} = I'_o\, e^{\frac{\kappa}{C_{tot}\,V_T}\,\sum_i^N C_i\,V_i}\, e^{\frac{\kappa\, C_{fd}\, V_d}{C_{tot}\, V_T}} \tag{3.7}$$

with I'_o constant, $C_{tot} = \sum_i C_i + C_{fd}$ the total capacitance of the floating gate, C_i the capacitance of input node i, C_{fd} the parasitic overlap capacitance floating gate-drain and N the amount of input nodes. By comparing equations 3.6 and 3.7 for floating gate transistors with one constant reference input V_{ref} on the capacitance C_{ref} and one input V the source-drain current can be written as

$$I_{sd} = I_o^{fg}\, e^{\frac{\kappa_{fg}\, V}{V_T}} \tag{3.8}$$

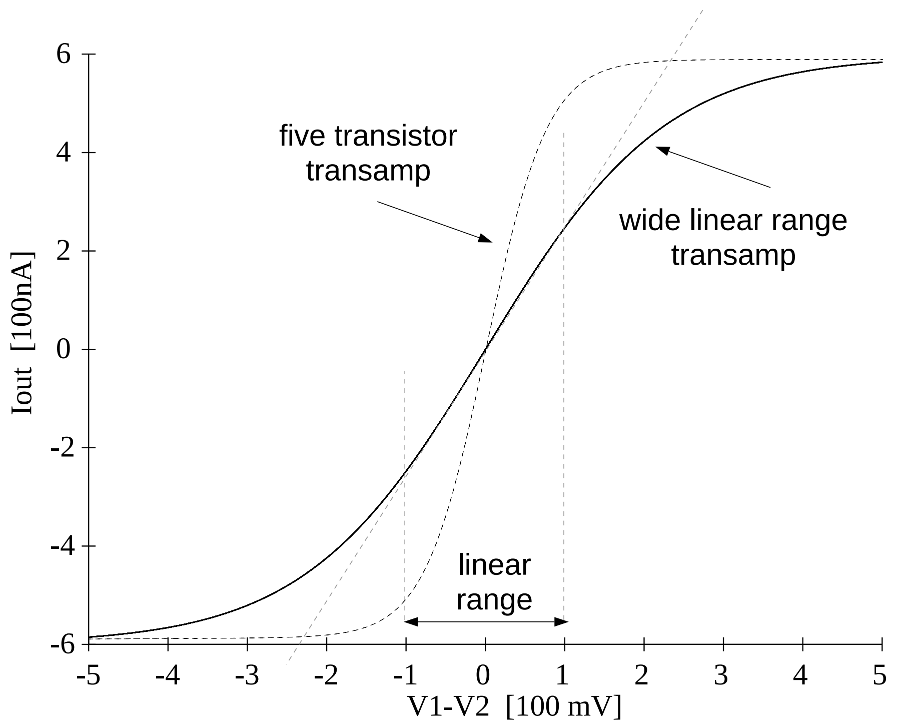


Figure 3.7: Analog simulation of wide linear range transconductance amplifier.

with the definition $\kappa_{fg} := \kappa \frac{C}{C+C_{ref}}$ and neglecting C_{fd}. Therefore the input-output characteristic for the floating gate transamp in figure 3.6 can simply be rewritten as

$$I_{out}^{fg} = I_b \tanh \frac{\kappa_{fg}\,(V_1 - V_2)}{2\,V_T}. \tag{3.9}$$

Linearizing the $\tanh$ for small arguments yields to the finding, that the slope of the floating-gate transamp is by a factor

$$\frac{C}{C + C_{ref}} \tag{3.10}$$

smaller the regular transamp. This means that the linear range of the regular transamp of $\approx 100\,mV$ can simply be increased by using the adequate capacitive divider ratio given by equation 3.10. It has to be kept in mind, though, that parasitic overlap capacitances will generally further reduce the influence of the control gate on the floating gate transistor, thus the linear range will be larger than suggested by equation 3.10.

It is possible to simulate floating gate devices with the software simulation tool *analog*. The trick is to set up the circuit and then connect the ground symbol to all floating gate at the same time, simulating charge equalization which is done in practice with UV light. After the ground symbols have been removed the circuit should simulate correctly. In Figure 3.7 the simulation result for a wide linear range transamp with capacitor sizes $C = 6 \times 8$ and $C_{ref} = 9 \times 8$ is shown, no parasitic capacitances are taken into account. The linear range in this case is increased by a factor of 2.5 as compared to the standard transamp.

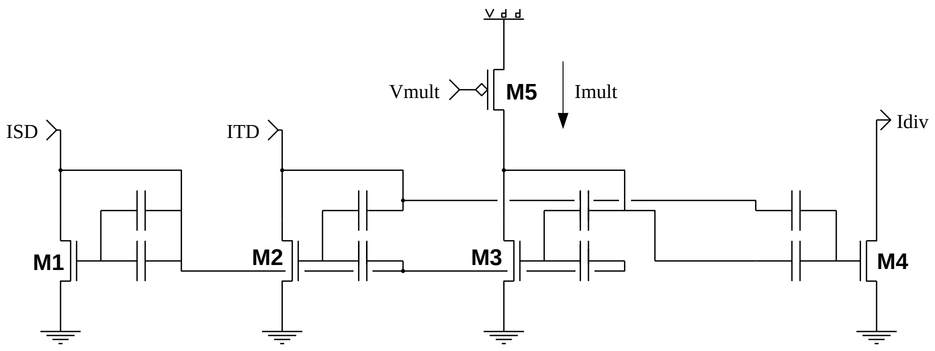


Figure 3.8: νMOS division circuit. This circuit can be layed out very compact and computes $I_{div} = I_{mult}\, I_{TD}/I_{SD}$ accurately as long as all currents stay sub-threshold.

The parasitic overlap capacitance between floating gate and drain C_{fd} can cause the spatial derivative circuit to deviate from the behavior described above. The reason is, that with changes in V_{out} through C_{fd} the floating gate of transistor M2 is influenced differently from the floating gate of transistor M1, thus changing the current I_{out}. The parasitic capacitance C_{fd} is a problem common to all floating gate translinear circuits and makes it necessary to pin the drain voltages as good as possible. *Analog* simulations showed that the difference in the drain voltages of M1 and M2 can be as high as 500mV.

3.3.5 Division circuit

The circuit for dividing the temporal by the spatial derivative (see figure 3.8) was developed following the translinear design principles of [MDHM96]. The circuit involves only 4 transistors $M1$ to $M4$ and one current source $M5$. The output current I_{div} is computed from I_{TD}, I_{SD} and a reference current I_{mult} such that

$$I_{div} = I_{mult}\, \frac{I_{TD}}{I_{SD}}. \tag{3.11}$$

This equation holds as long as all currents stay sub-threshold, i.e. $< 100\, nA$. All capacitors with floating gates were designed as small as $6 \times 6\, \lambda$, because the capacitor matching is known to be sufficient even for this small area size. A different circuit with 4 transistors plus current source but without floating gates for the same operation was proposed in [ABP+91].

3.3.6 Sign correction

After the temporal derivative has been divided by the spatial derivative, the sign has to be corrected, because the division could only be performed on unidirectional currents. For this purpose a high gain differential amplifier is used (see figure 3.9), which receives the intensity input from left and right neighbor, thus computing the sign of the spatial derivative. Depending on the sign the current I_{div} from the division circuit is either added to or subtracted from the output wire, which is held at a constant potential of 4V. Through this wire the output is scanned off the chip.

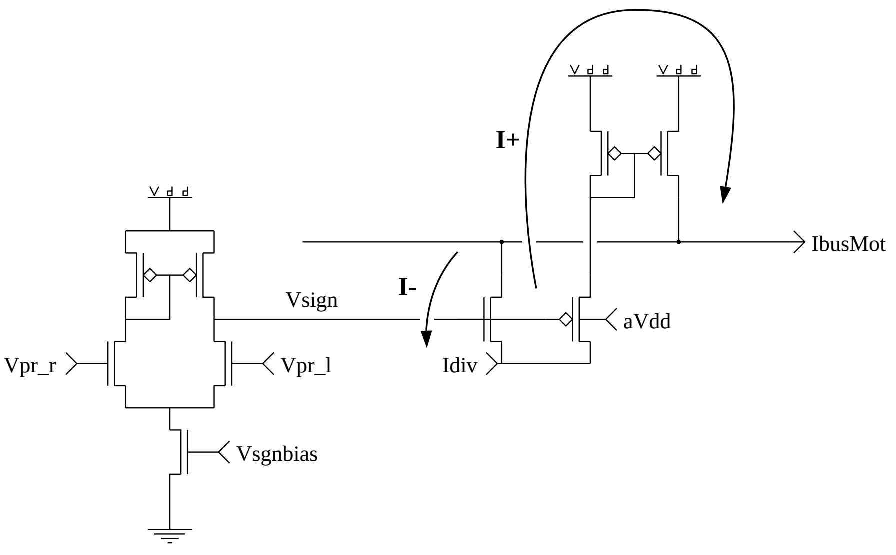


Figure 3.9: Sign correction circuit. The light intensities of left and right neighbor are compared in a high gain differential amplifier. The voltage V_{sign} is at $5\,V$ if $V_{pr\_r} > V_{pr\_l}$ and at GND for $V_{pr\_r} < V_{pr\_l}$, thus driving one of the switching transistors such that the current I_{div} is either added to or subtracted from the bus wire I_{busMot}.

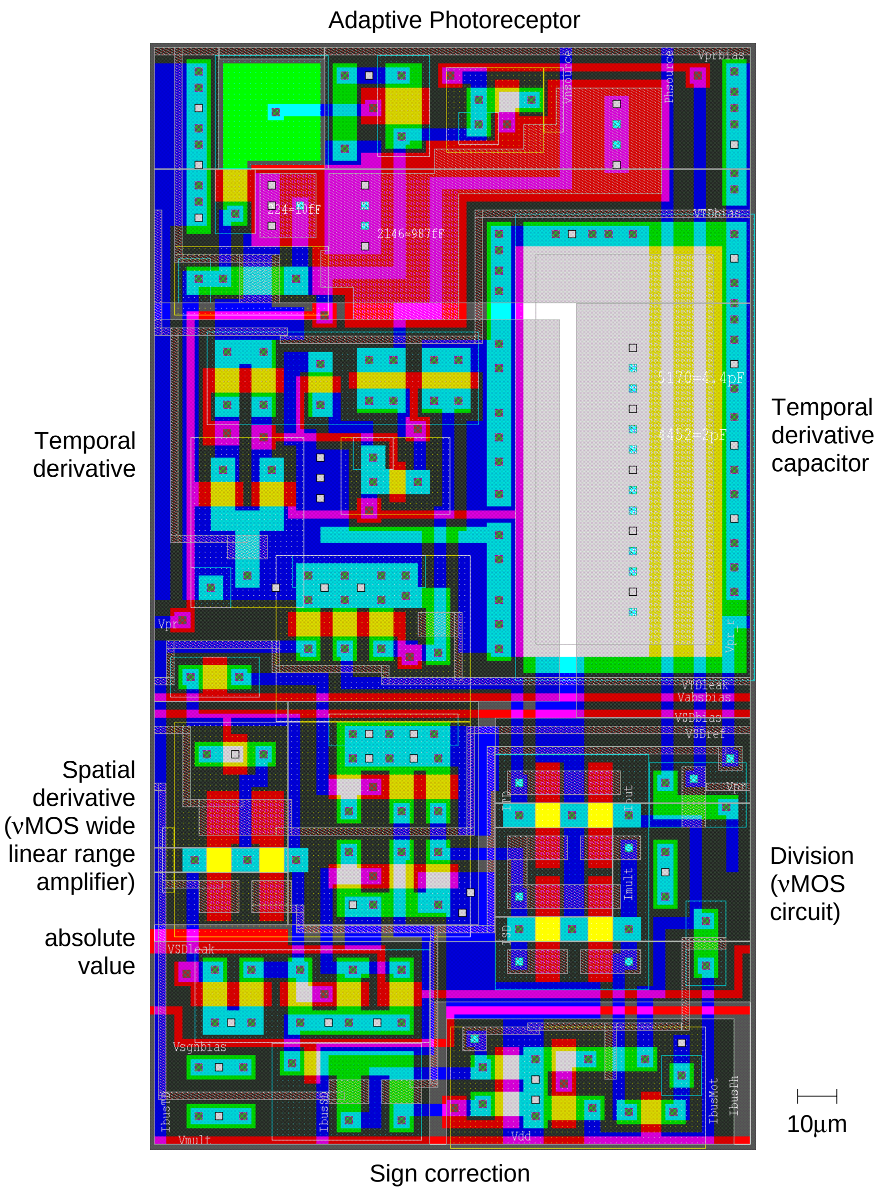


Figure 3.10: Pixel layout of CMotion1d. The data processing stream goes top to bottom. On the upper left corner the $24\mu m \times 24\mu m$ active area is visible where the light is collected. The large rectangular area to the right is the capacitor for the temporal derivative. On the left are temporal and spatial derivative circuits, on the lower right part the νMOS division circuit is located. On the bottom the sign is corrected and the velocity signal in form of a current is passed out. The cell size in $2.0\mu m$ technology is $147\mu m \times 270\mu m$.

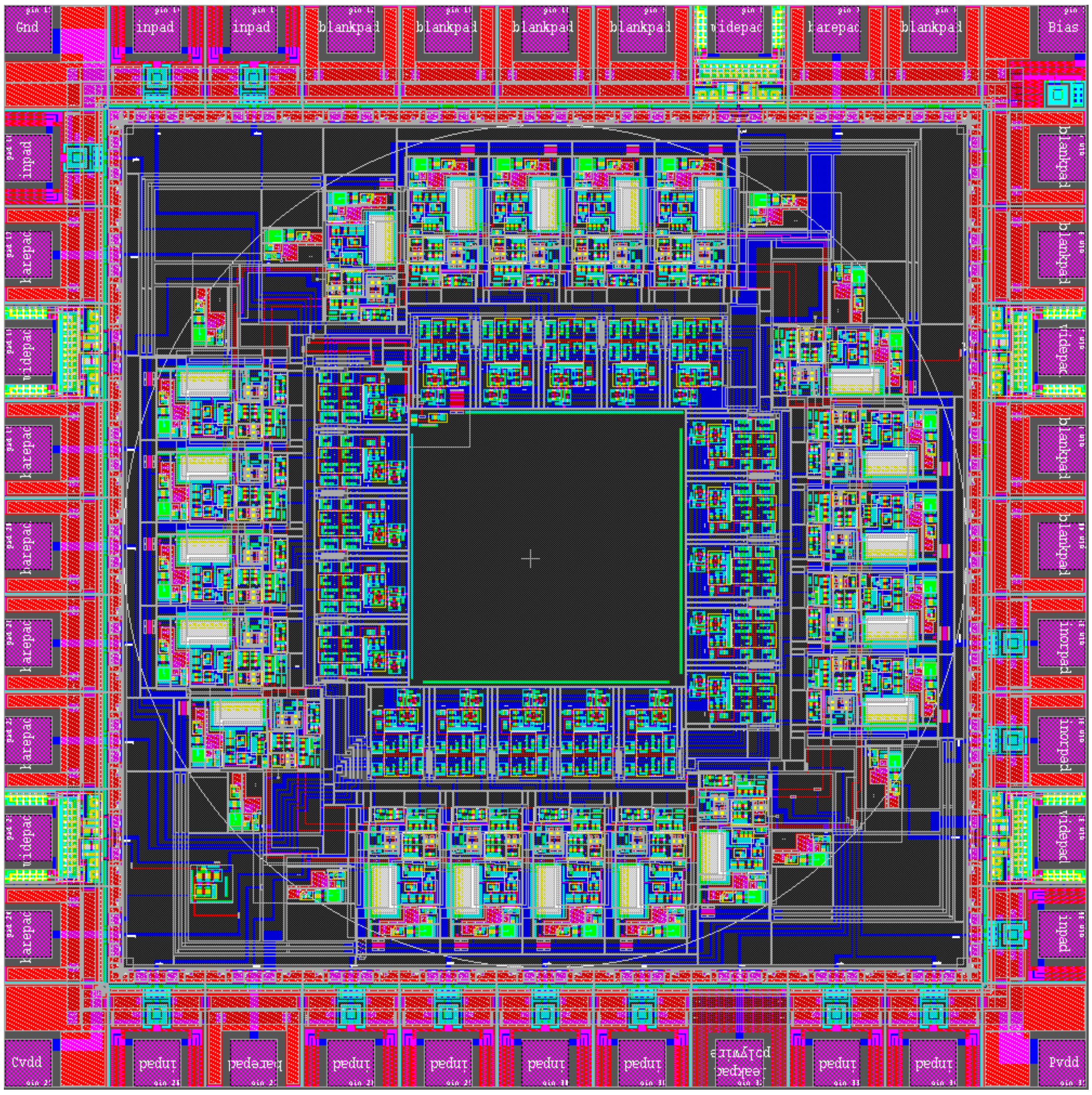


Figure 3.11: Layout of the CMotion1d velocity sensor. A linear array of elementary velocity sensors is arranged in a circle such that rotational velocity can be detected. Any three adjacent pixels interact and implement the gradient method for velocity computation. The temporal derivative is computed locally, the spatial derivative is computed by taking the difference between the light intensity of left and right neighbor.

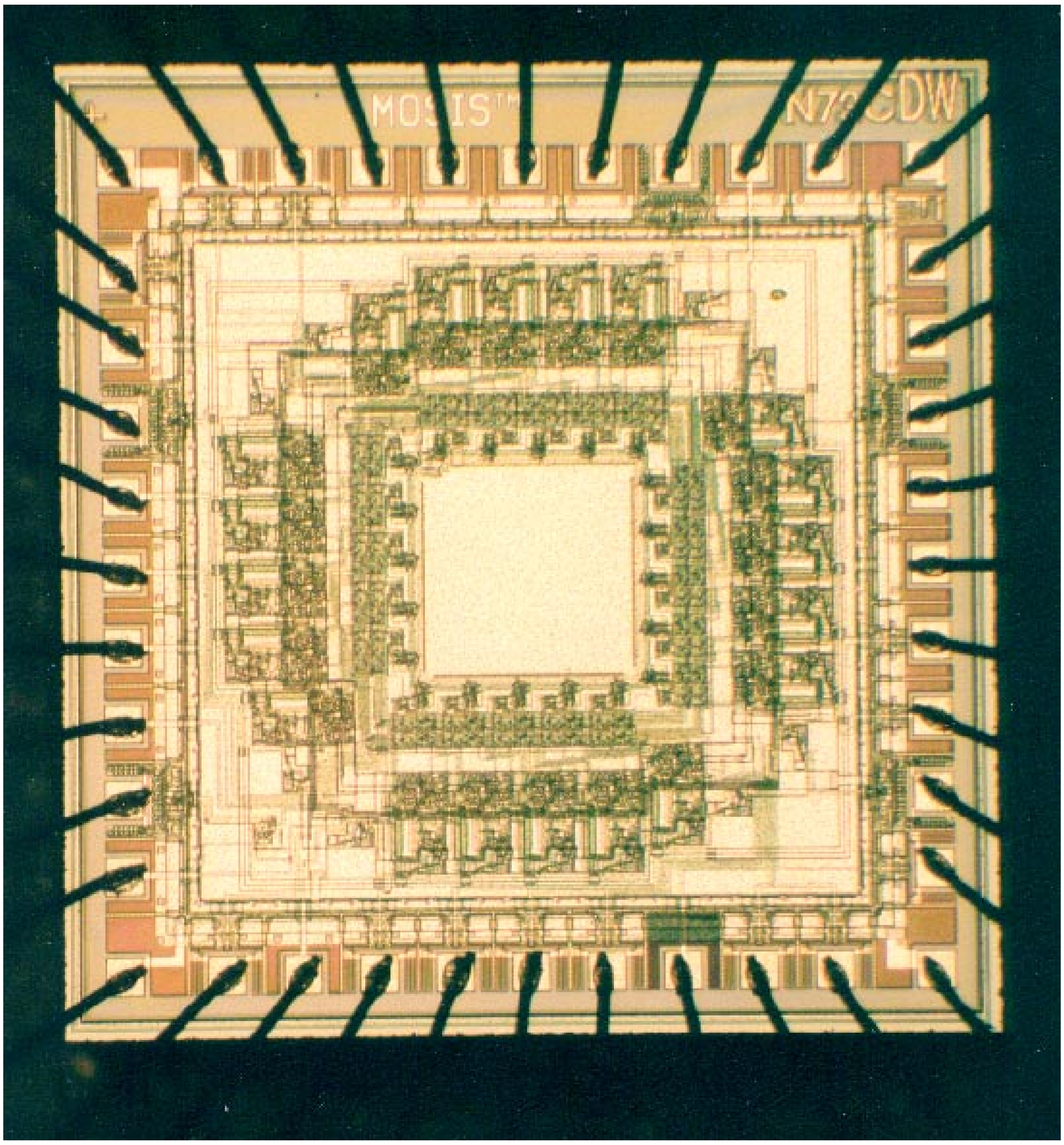


Figure 3.12: Micro photograph of the CMotion1d chip.

3.4 Fundamental Limits

For the velocity signal obtained with the 1-D gradient method from the theory the following behavior is expected:

- Linearity with the stimulus velocity
- Independency of the input signal spatial frequency content
- Independency of the input signal contrast
- Time independence
- $1/\cos\varphi$ behavior for input signals in direction φ

As will be shown in this section, for every hardware implementation of the gradient method necessarily new effects will be introduced:

- All active pixel sensor implementations, where imaging and computation is performed on a 2-D chip surface, necessarily have a fill factor smaller than 100%. Therefore, like in the CMotion1d sensor, computation can be done continuously in time, but only discretely in space. As a result temporal aliasing cannot occur, but a dependency on spatial frequency for any computation which approximates an infinitesimal computation in space, like the velocity computation with the gradient method, is expected.
- Contrast independency of the velocity signal can only be expected for input signals well above the noise level. Very low contrasts in hardware as well as in software implementations of the gradient method have to be treated separately. The sensor should be implemented such that for vanishing contrasts the velocity output goes to zero.
- Because the division by zero problem has to be dealt with, deviations from the optimal sensor output for very low temporal and spatial derivatives are expected.
- In a hardware implementation a linear velocity output is expected only for a limited range of stimulus velocities. For high stimulus speeds ranges of operation of circuits in the sensor will be exceeded and low pass filter time constants will become important.
- DC offsets and gain offsets are to be expected.

3.4.1 Effects of a discrete spatial derivative and spatio-temporal filtering

Let the spatio-temporal light intensity distribution moving with velocity v on the sensor surface be

$$I(x,t) = I(x - v \cdot t) \tag{3.12}$$

where $x \in \mathbb{R}^1$ is the 1-D location on the sensor surface and $t \in \mathbb{R}^1$ the time.

Assume a separable spatio-temporal filter with the respective Fourier representation

$$g(t) \rightarrow \hat{g}(\omega) \tag{3.13}$$

$$h(x) \rightarrow \hat{h}(k) \tag{3.14}$$

then the filtered signal can be written as

$$(I * g * h)(x,t) = \frac{1}{2\pi}\int\int e^{-i\omega t}\, e^{ikx}\, \hat{I}(\omega,k)\, \hat{h}(k)\, \hat{g}(\omega)\, d\omega dk. \tag{3.15}$$

The temporal derivative of this filtered signal becomes

$$I_t(x,t) := \frac{\partial I(x,t)}{\partial t} = -\frac{i}{2\,\pi}\int\int \omega\, e^{-i\,\omega\,t}\, e^{i\,k\,x}\, \hat{I}(\omega,k)\, \hat{h}(k)\, \hat{g}(\omega)\, d\omega\, dk. \tag{3.16}$$

A) Infinitesimal spatial derivative

If the spatial derivative were computed infinitesimally on chip then

$$I_x(x,t) := \frac{\partial I(x,t)}{\partial x} = \frac{i}{2\,\pi}\int\int k\, e^{-i\,\omega\,t}\, e^{i\,k\,x}\, \hat{I}(\omega,k)\, \hat{h}(k)\, \hat{g}(\omega)\, d\omega\, dk. \tag{3.17}$$

and the velocity signal v_{out} would be, using $\omega = v\,k$

$$\begin{aligned} v_{out} &= -\frac{I_t(x,t)}{I_x(x,t)} = \\ &= \frac{\int\int \omega\, e^{-i\,\omega\,t}\, e^{i\,k\,x}\, \hat{I}(\omega,k)\, \hat{h}(k)\, \hat{g}(\omega)\, d\omega\, dk}{\int\int k\, e^{-i\,\omega\,t}\, e^{i\,k\,x}\, \hat{I}(\omega,k)\, \hat{h}(k)\, \hat{g}(\omega)\, d\omega\, dk} = \\ &= v \end{aligned} \tag{3.18}$$

This result shows that:

- The reported velocity does not depend on the stimulus signal.
- The reported velocity is time independent.
- The reported velocity is not affected by neither temporal nor spatial filtering on the input signal.

B) Discrete spatial derivative

In a hardware implementation the spatial derivative can be computed by a pixel at $x = 0$ from the difference of the light intensity of the right neighbor ($x = \Delta$) and left neighbor ($x = -\Delta$):

$$\begin{aligned} I_\Delta(t) &:= \frac{I(\Delta,t) - I(-\Delta,t)}{2\,\Delta} = \\ &= \frac{1}{2\,\pi}\int\int e^{-i\,\omega\,t}\, \hat{I}(\omega,k)\, \hat{h}(k)\, \hat{g}(\omega)\, (\frac{e^{i\,k\,\Delta} - e^{-i\,k\,\Delta}}{2\,\Delta})\, d\omega\, dk = \\ &= \frac{1}{2\,\pi}\int e^{-i\,v\,k\,t}\, \hat{I}(k)\, \hat{h}(k)\, \hat{g}(v\,k)\, \frac{\sin(k\,\Delta)}{\Delta}\, dk \end{aligned} \tag{3.19}$$

if the ω integration is carried out with $\hat{I}(\omega, k) = \hat{I}(k)\, \delta(\omega - v\, k)$.

On chip the velocity signal at position $x = 0$ is then obtained by dividing the temporal and the discrete spatial derivatives

$$\begin{aligned} v_{out} &= -\frac{I_t(0,t)}{\frac{I(\Delta,t)-I(-\Delta,t)}{2\,\Delta}} = \\ &= v\, \frac{\int k\, e^{-i\,v\,k\,t}\, \hat{I}(k)\, \hat{h}(k)\, \hat{g}(v\, k)\, dk}{\int e^{-i\,v\,k\,t}\, \frac{\sin(k\,\Delta)}{\Delta}\, \hat{I}(k)\, \hat{h}(k)\, \hat{g}(v\, k)\, dk} \end{aligned} \tag{3.20}$$

From this result several facts can be learned:

- The reported velocity output v_{out} in general is time dependent: $v_{out} = v_{out}(t)$.
- v_{out} is dependent on the spatial frequency content $\hat{I}(k)$ of the input intensity distribution.
- Spatial filtering of the input light intensity with a filter $\hat{h}(k)$ will affect v_{out}. For example spatial blurring due to the lens

$$h(x) = \frac{1}{\sqrt{2\,\pi\,\sigma^2}}\, e^{-x^2/2\,\pi\,\sigma^2} \rightarrow \hat{h}(k) = e^{-k^2\,\sigma^2/2} \tag{3.21}$$

  will result in different reported velocities for different blurring strengths σ.
- Also temporal filtering of the input light intensity will affect v_{out}. With a first order low pass filter approximation of the adaptive photoreceptor

$$g(t) = \frac{1}{\tau}\, e^{-t/\tau} \rightarrow \hat{g}(\omega) = \frac{1}{1 + i\,\omega\,\tau} \tag{3.22}$$

  changing the receptor bias will also change the velocity output.
- Even when there is no temporal filtering ($\hat{g}(\omega) = 1$) the spatial filter affects v_{out}. Similarly v_{out} is affected by temporal filtering alone ($\hat{h}(k) = 1$). Even for no filtering at all v_{out} remains time dependent.
- Equivalent to the dependency of v_{out} on the spatial frequency content of the signal v_{out} also is dependent in the pixel spacing Δ.

3.4.2 Temporal low pass filtering in the photoreceptor

If the stimulus is assumed to be a pure sine wave

$$I(x,t) = I_o\, \sin(\omega\, t - k\, x). \tag{3.23}$$

and the photoreceptor response is modeled by a first order temporal low pass filter

$$g(t) = \frac{1}{\tau}\, e^{-t/\tau} \rightarrow \hat{g}(\omega) = \frac{1}{1 + i\,\omega\,\tau} \tag{3.24}$$

then the signal from the photoreceptor becomes by virtue of equation 3.15 with$\hat{h}(k) = 1$

$$\begin{aligned} I^{LP}(t) &= Re\,(I(t) * g(t)) = \\ &= I_o\,\frac{\sin(\omega\,t - k\,x) + \omega\,\tau\,\cos(\omega\,t - k\,x)}{1 + \omega^2\,\tau^2} \end{aligned} \tag{3.25}$$

which is reasonable, because for $\tau \to 0$ the signal $I(x,t)$ is obtained and for $\tau \to \infty$ the filtered signal goes to zero.

The temporal derivative circuit computes from the low pass filtered photoreceptor signal at the pixel location $x = 0$

$$I_t^{LP}(t) := \left.\frac{\partial I^{LP}(t)}{\partial t}\right|_{x=0} = I_o\,\omega\,\frac{\cos(\omega\,t) + \omega\,\tau\,\sin(\omega\,t)}{1 + \omega^2\,\tau^2}. \tag{3.26}$$

The spatial derivative is obtained from the difference of the photoreceptor signals from the left and right pixels ($x = -\Delta$ and $x = \Delta$):

$$I_\Delta^{LP}(t) := \frac{I^{LP}(\Delta,t) - I^{LP}(-\Delta,t)}{2\,\Delta} = I_o\,\frac{\sin(k\,\Delta)}{\Delta}\,\frac{\cos(\omega\,t) + \omega\,\tau\,\sin(\omega\,t)}{1 + \omega^2\,\tau^2}. \tag{3.27}$$

To get rid of the time dependence the extremes of $I_t^{LP}(t)$ and $I_x^{LP}(t)$ are derived as

$$\begin{aligned} I_t^{LP}(t)\ \&\ I_\Delta^{LP}(t) \overset{!}{=} Ext \quad &\Leftrightarrow \quad \frac{\partial}{\partial t}(\cos(\omega\,t) + \omega\,\tau\,\sin(\omega\,t)) = 0 \\ &\Leftrightarrow \quad \omega\,t = \arctan(\omega\,\tau) \end{aligned}$$

where the extremes are given by

$$I_t^{LP} = I_o\,\frac{\omega}{\sqrt{1 + \omega^2\,\tau^2}} \tag{3.28}$$

$$I_\Delta^{LP} = I_o\,\frac{\sin(k\,\Delta)}{\Delta}\,\frac{1}{\sqrt{1 + \omega^2\,\tau^2}}\,,\ \omega = v\,k. \tag{3.29}$$

In the CMotion1d sensor for debugging reasons the temporal and spatial derivatives are available from every pixel. Therefore the values of I_t^{LP} and I_Δ^{LP} can be obtained from experiment with stimuli of different velocities. The above derived expressions will be used to fit the data, and time constants τ will be obtained from the temporal and spatial derivative measurement. As will be shown, both expressions will fit the data very well, and the time constants obtained from temporal and spatial derivatives will differ slightly.

3.4.3 Velocity output for pure sine wave stimuli

In general for a sinusoid input light intensity distribution with fixed spatial frequency k the velocity output v_{out} becomes

$$v_{out} = v\,\frac{k\,e^{-ivkt}\,\hat{I}(k)\,\hat{h}(k)\,\hat{g}(v\,k)}{e^{-ivkt}\,\sin(k\,\Delta)\,\hat{I}(k)\,\hat{h}(k)\,\hat{g}(v\,k)}\,\Delta =$$

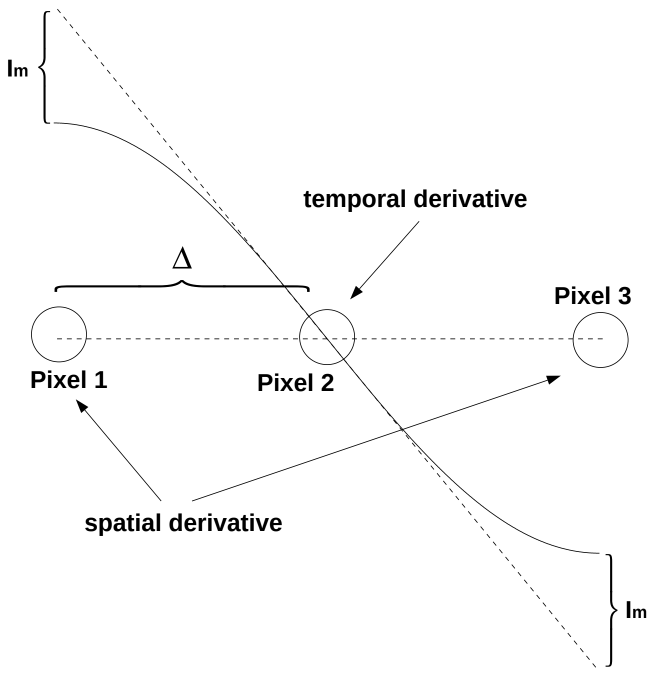


Figure 3.13: Reason for spatial frequency dependence of velocity output. The larger the spatial frequency of the stimulus, the stronger the deviation between discrete and continuous spatial derivative. This is because the discrete intensity difference between pixel 1 and pixel 3 is missing the intensity I_m twice. Since the temporal derivative is taken at the position of pixel 2, the resulting velocity output will become larger with higher spatial frequencies.

$$= \quad v \, \frac{k\,\Delta}{\sin(k\,\Delta)} \tag{3.30}$$

and it can be observed that

- v_{out} does neither depend on spatial nor temporal filtering.
- v_{out} is not dependent on time.
- v_{out} is significantly dependent on the spatial frequency k.

The spatial frequency dependence of the velocity output v_{out} can be understood by the fact that the temporal derivative is taken at the central pixel whereas the spatial derivate is computed as difference signal from the two finitely spaced neighboring pixels. Therefore even though the temporal derivative is correct, the spatial derivative will be too small due to the missing intensities I_m, as depicted in Figure 3.13.

For spatial frequencies $k > \pi/\Delta$ spatial aliasing is observed. Even though the temporal derivative is still computed correctly, the spatial derivative goes through zero (see Figure 3.14) and reverses sign. Therefore for $k \to \pi/\Delta$ the motion signal increases and at $k = \pi/\Delta$ reverses sign, too. See

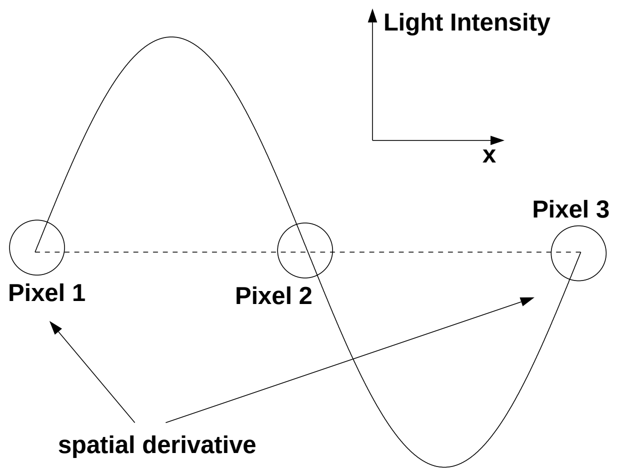


Figure 3.14: The aliasing condition. As the wavelength of the stimulus equals twice the pixel spacing Δ, the discrete spatial derivative vanishes and the velocity output is infinite.

Figure 3.15 for a simulation of the velocity signal for the discrete gradient method for different spatial frequencies. In the results section 3.7 it is demonstrated that aliasing is observed in the hardware implementation.

3.4.4 Temporal low pass filtering in the derivative circuits

Up to now only the temporal low pass filter behavior of the front end photoreceptor has been considered. It was found that although temporal and spatial derivatives are affected, the velocity output is still linear with stimulus speed for sine wave stimuli. This is because of the division operation.

It will now be shown, that the velocity output becomes nonlinearly dependent on the stimulus speed if there are also time constants associated with the temporal and spatial derivative circuits and these time constants are mutually different. For that purpose a model of the CMotion1d sensor containing three temporal low pass filter time constants is proposed, one for the photoreceptor and two possibly different ones for both derivatives. This model is depicted in Figure 3.16.

Consider the input light intensity distribution $I(x,t)$ and two temporal low pass filters, $\hat{g}_1(\omega)$ for the photoreceptor and $\hat{g}_2(\omega)$ for either one derivative circuit, with time constants τ_1 and τ_2 respectively. With $\mathcal{D}$ a derivative operator, the filtered derivative output is expressed by

$$\mathcal{D}(I * g_1) * g_2 = (\mathcal{D}\, I) * (g_1 * g_2) \tag{3.31}$$

The combined filter $(g_1 * g_2)$ can be obtained in Fourier representation simply by multiplication:

$$\widehat{(g_1 * g_2)} = \frac{1}{1 + i\,\omega\tau_1}\,\frac{1}{1 + i\,\omega\tau_2} \tag{3.32}$$

With the absolute value

$$\left|\widehat{(g_1 * g_2)}\right| = \frac{1}{\sqrt{1 + \omega^2(\tau_1^2 + \tau_2^2) + \omega^4\,\tau_1^2\,\tau_2^2}} \tag{3.33}$$

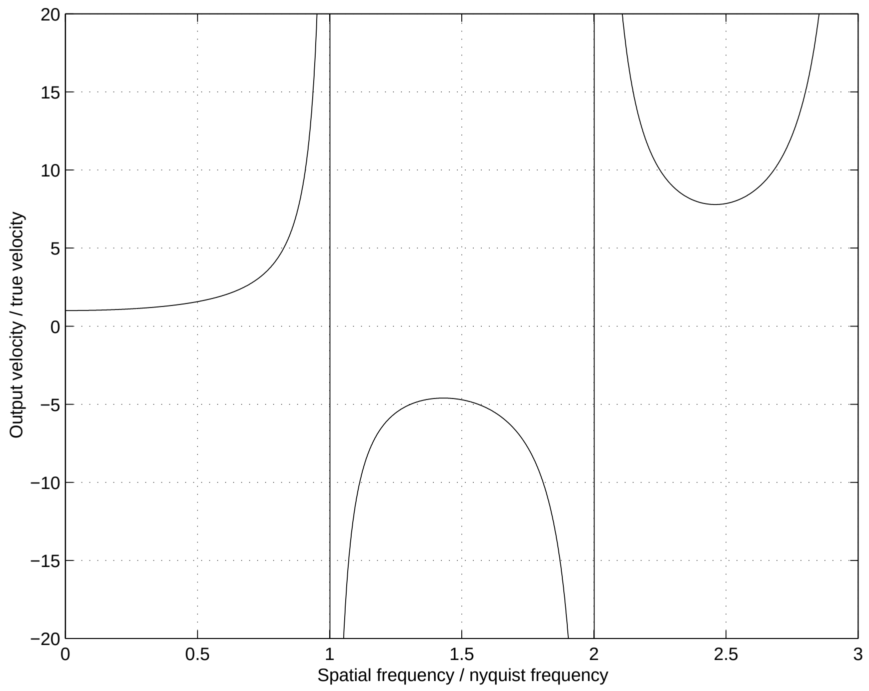


Figure 3.15: Spatial frequency dependence of velocity output. The function which is plotted is: $n\,\pi / \sin(n\pi)$ where $n = k/k_{nyq}$, $k_{nyq} = \pi/\Delta$ being the Nyquist frequency and Δ the inter pixel spacing

the maximum values of the temporal and spatial derivative circuits become

$$I_t^{LP} = I_o \frac{\omega}{\sqrt{1 + \omega^2(\tau_1^2 + \tau_{TD}^2) + \omega^4\,\tau_1^2\,\tau_{TD}^2}} \tag{3.34}$$

$$I_\Delta^{LP} = I_o \frac{\sin(k\,\Delta)}{\Delta} \frac{1}{\sqrt{1 + \omega^2(\tau_1^2 + \tau_{SD}^2) + \omega^4\,\tau_1^2\,\tau_{SD}^2}} \tag{3.35}$$

where τ_{TD} and τ_{SD} denote the time constants for the temporal and spatial derivative circuit, respectively. It can be seen that for vanishing τ_{TD} and τ_{SD} equations 3.28 and 3.29 are gained back for only low pass filtering in the photoreceptor.

With these results the peak motion output is given by

$$v_{out} = \frac{I_t^{LP}}{I_\Delta^{LP}} = v\,\frac{k\,\Delta}{\sin(k\,\Delta)}\,\frac{\sqrt{1 + \omega^2(\tau_1^2 + \tau_{SD}^2) + \omega^4\,\tau_1^2\,\tau_{SD}^2}}{\sqrt{1 + \omega^2(\tau_1^2 + \tau_{TD}^2) + \omega^4\,\tau_1^2\,\tau_{TD}^2}} \tag{3.36}$$

It can be seen that the velocity output v_{out} in the case of mutually different low pass filtering time constants τ_{TD} and τ_{SD} is not linear in the stimulus velocity $v = \omega/k$. If the time constant in the temporal derivative circuit is larger, then the velocity output will be compressively nonlinear

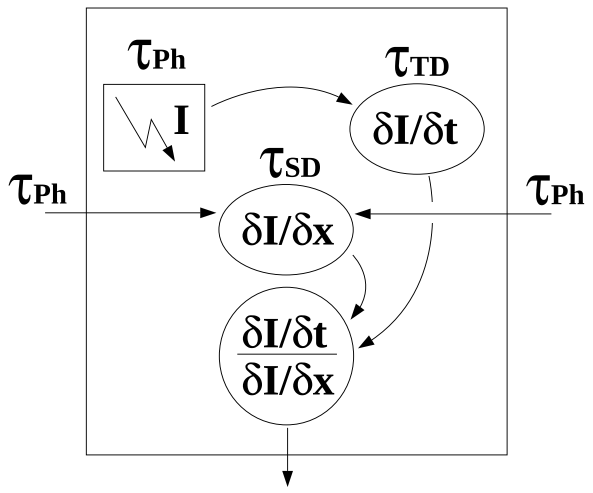


Figure 3.16: Model of the CMotion1d sensor with time constants. Different time constants are assumed to be associated with the photoreceptor and both derivative circuits. Since there are two separate pathways for temporal and spatial derivative up to the division operation, different time constants τ_{TD} and τ_{SD} cause nonlinearity of the velocity output for different stimulus velocities.

in the stimulus velocity. This equation will be used to fit the velocity data obtained with the CMotion1d sensor and to asses to what extent different low pass filter time constants play a role in the nonlinearity of the velocity output.

3.4.5 Above threshold transistor operation

A further contribution to a nonlinear velocity output shall be mentioned here. The νMOS circuit used for the division in the CMotion1d sensor is operating correctly only for subthreshold currents, where the source-drain current I_{sd} is exponentially dependent on the gate voltage V_g. For larger currents I_{sd} is approximately quadratically dependent on the gate voltage. Consider the output transistor M4 in the νMOS division circuit in Figure 3.8. If both its inputs are such that M4 goes above threshold, then a further increase in one of its input voltages, for example from the temporal derivative, will only increase its output current quadratically instead of exponentially. Therefore a compressive nonlinearity would be measured.

3.4.6 The division by zero problem

As already mentioned before, in the CMotion1d sensor constantly a current c_t can be subtracted from the temporal derivative and a different current c_Δ can be added to the spatial derivative:

$$v_{out} = -\frac{\max(0, I_t - c_t)}{I_\Delta + c_\Delta}. \tag{3.37}$$

The numerator is implemented through a bypass transistor in the temporal derivative circuit. The stronger the bias V_{TDleak}, the more current flows through this transistor instead of being reported as

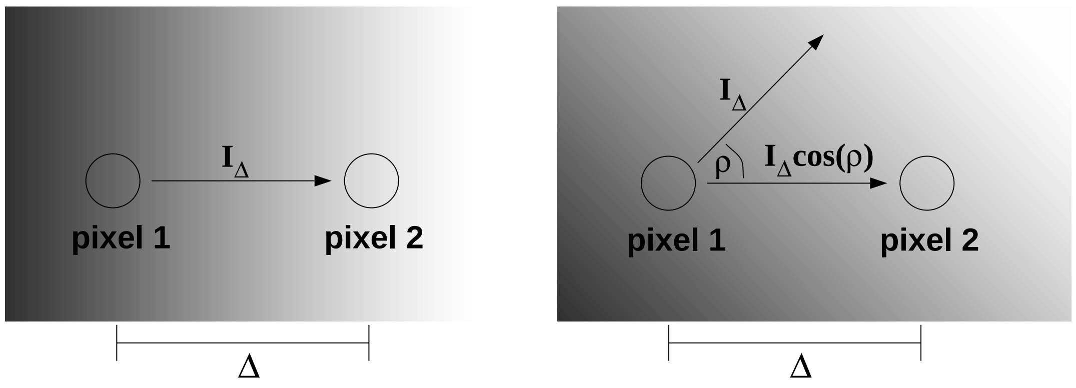


Figure 3.17: Angular dependency of the spatial derivative $I_\Delta(\varphi) = I_\Delta \cos(\varphi)$. φ is the angle in which the intensity gradient is oriented, I_Δ is the gradient for $\varphi = 0$ and Δ is the pixel spacing. On the left the case for $\varphi = 0^o$ is depicted, on the right side $\varphi = 45^o$. It is obvious that for $\varphi = 90^o$ and $\varphi = 270^o$ the spatial derivative vanishes and thus the velocity output in the gradient method goes to ∞.

temporal derivative. The denominator is implemented through a leakage transistor in the absolute value circuit. The stronger the bias V_{SDleak} the more current will be added to the spatial derivative. Effects of this procedure are observed and discussed in the results sections 3.5 and 3.7.

3.4.7 Orientation tuning curve

If the intensity gradient is assumed to be linear between the pixels which compute the spatial derivative then the spatial derivative $I_\Delta(\varphi)$ goes like the cosine of the angle φ of the gradient. The expected angular dependent velocity output for the gradient method is then

$$v_{out}(\varphi) = -\frac{I_t}{I_\Delta(\varphi)} = -\frac{I_t}{I_\Delta}\frac{1}{\cos(\varphi)} = v_{out}\frac{1}{\cos(\varphi)} \tag{3.38}$$

where I_t is the temporal derivative, I_Δ is the discrete spatial derivative and v_{out} is the velocity output for $\varphi = 0$. In the results section 3.9 the orientation tuning curve of the CMotion1d sensor is measured. Equation 3.38 will be confirmed.

3.5 Results: Sample Outputs

In this section sample outputs of the CMotion1d sensor are demonstrated. Photoreceptor voltage, temporal and spatial derivative and velocity output are recorded over time and taken off the oscilloscope. Two two-channel scopes were programmed to act like one four channel scope. The sample outputs are to demonstrate the operation of the sensor and to explain the influence of some important bias voltages.

Figure 3.18 demonstrates the operation of the CMotion1d sensor. A sine wave was used as stimulus and the outputs were recorded over one cycle. The biases were set for an optimal operation. The top trace shows the absolute value of the spatial derivative together with the theoretical function (dashed line). It can be seen that the absolute value circuit introduces asymmetry in that the first half wave of the sine is smaller than the second half wave. Since the temporal derivative due to the way how it is implemented is computed only for decreasing intensities, the effect for the velocity output is that for one direction-of-motion the velocity signal is larger than for the other one.

The second trace shows the output of the temporal derivative circuit together with the theoretical expectation. It can be seen that only one sign of the temporal derivative of the intensity signal is computed. During the time of increasing intensities the temporal derivative output is zero. The onset of the temporal derivative on decreasing intensities is delayed due to the crossover distortion. This effect is caused by the delayed onset of current flow through one of the two transistors at the output of the differential amplifier in the temporal derivative circuit. The time derivative signal shows some oscillations which are due to the feedback in the circuit. The amplitude of these oscillations increases the higher the cutoff frequency of the photoreceptor is set, i.e. the higher frequency components are present in the input signal. Also for brighter light or wider open aperture on the lens the oscillations tend to increase. It was experimentally verified, though, that even for large oscillations in the temporal derivative the smoothed temporal derivative and velocity signals were still correct.

The third trace shows spatial and temporal derivatives together. It is crucial for the gradient method that the time course of those two values is in sync. As can be seen this is given.

On the fourth trace, finally, the velocity output is displayed together with the photoreceptor signal. This trace shows one of the most exciting findings of this work. It proves the validity of the implementation of the gradient method: Through the division operation in real time the velocity signal is computed as a constant during the time where the temporal derivative signal is present, thus a square wave with duty cycle of approximately 1/2 is generated from the two sinusoidal signals. The heights of the square is proportional to the stimulus velocity, as will be shown later. This height is taken as signal for the experiments presented in the subsequent results sections 3.6 to 3.11. For the length of the signal it becomes clear that for this sensor the velocity output is computed only during times where a stimulus is present, i.e. there are temporal and spatial derivatives in the visual scene. There is no artificial persistence time introduced as for example in the FTC sensor. The overlaid photoreceptor output shows that the velocity signal is computed during falling intensities.

In the following figures is is shown in detail, how the velocity output is computed and what effects can be observed for different bias settings. All figures show on the top traces the temporal and spatial derivatives overlaid, and on the bottom traces the measured velocity output together with the photoreceptor output. All experiments were taken under the same conditions, except for the change in the bias setting that was investigated. Explanations to the figures are given in the respective captions. First the division by zero problem is demonstrated, then several ways are pointed out to deal with that problem. The influence of the absolute value circuit is demonstrated as well as the effect of the photoreceptor bias.

Figure 3.26 finally clearly demonstrates that the velocity output of the division circuit can actually be reconstructed by dividing the measured temporal and spatial derivatives in the computer. This result is very interesting and has never been demonstrated for a dynamic system before. The reconstruction can only be positive because the measured derivative currents are absolute values. Therefore the peak at the end of the second square wave is positive instead of negative as in the original output. The division result was multiplied by 110 nA, which corresponds to the multiplication current in the division circuit set by V_{mult}.

Bias settings

In the following the bias settings for proper operation of the CMotion1d sensor are listed, together with some comments:

1. *aVdd=5V* Analog Vdd is driving the pixels and is kept separate from digital Vdd to avoid glitches from switching noise.
2. *dVdd=5V* Digital Vdd is driving the scanner circuitry.
3. *sVdd=12V* is the supply voltage to the operational amplifiers LM324. Some head room for the output voltages is needed because the reference voltages are at 4V.
4. $V_{TDbias} = 3.00V$ is the bias on the temporal derivative circuit. For weaker bias settings ($V_{TDbias} > 3.80V$) the derivative starts to oscillate, for stronger bias settings the cross over distortion becomes stronger.
5. $V_{nsource} = 190mV$ exponentially determines the amplification of the temporal derivative current. For very slow velocities $V_{nsource}$ can be increased. The temporal derivative current must not exceed $\approx 100\,nA$ in order to stay sub-threshold.
6. $V_{TDleak} = 3.80V$ is used to minimize the division by zero problem by subtracting a constant current off the temporal derivative.
7. $V_{SDbias} = 4.03V$ determines the amplification of the spatial derivative current.
8. $V_{SDleak} = 676mV$ adds a constant current to the spatial derivative. Used to minimize the division by zero problem.
9. V_{SDfg} is the voltage on the reference input to the floating gate of the spatial derivative circuit. V_{SDfg} can be set anywhere between 0V and 4V without changing the sensor performance.

10. $V_{abs} = 1.228V$ determines the DC level and shape of the rectified spatial derivative.
11. $V_{mult} = 4.20V$ determines the amplification of the velocity output current. Has to be set such that the output current stays sub-threshold.
12. $V_{sgn} = 600mV$ determines the speed with which sign changes in the output current can be performed.
13. $V_{pr} = 3.94V$ is the photoreceptor bias. The bias is low because a Poly2 gate is used in the layout for the bias transistor. If V_{pr} is biased stronger the photoreceptor low pass filter time constant decreases and the temporal derivative starts to oscillate.
14. $V_{TDref}, V_{SDref}, V_{Motref}$ are the potentials at which the output wires are held by the scanner. They are set to 4V because all outputs are generated by pFETs.

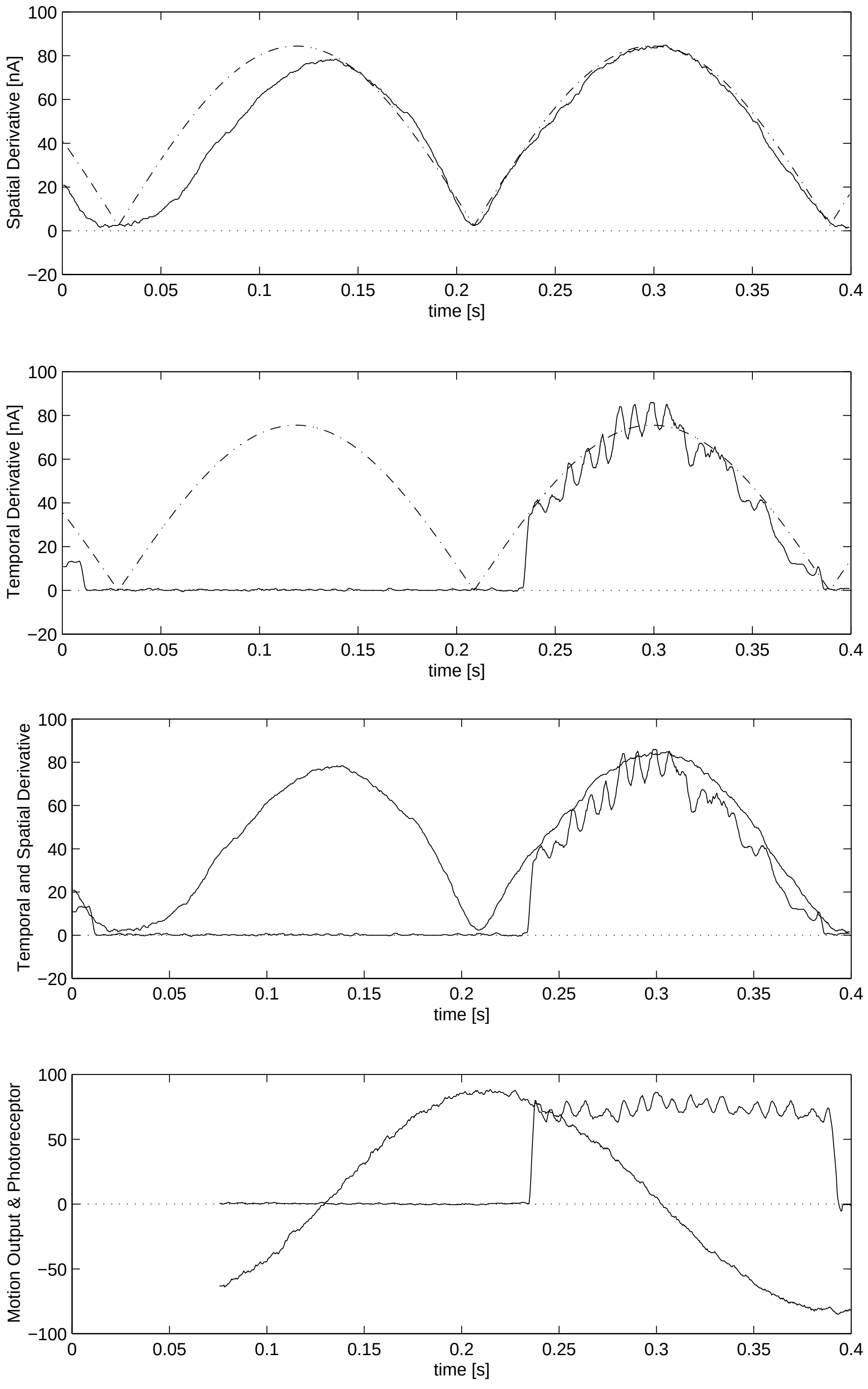


Figure 3.18: Time course of velocity signal generation.

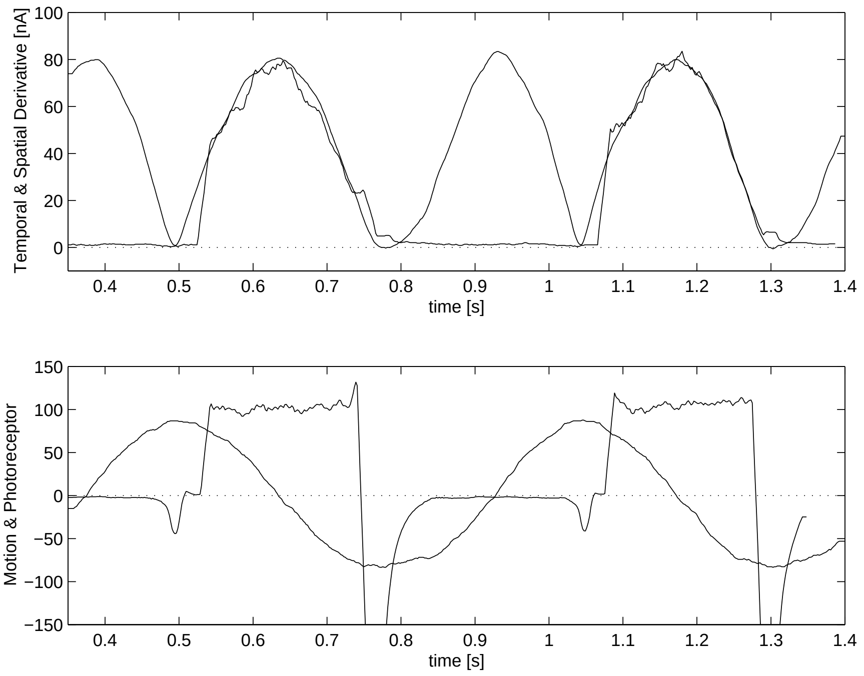


Figure 3.19: Division by zero problem. The velocity output (lower trace) grows very large for times when the spatial derivative is very small and the temporal derivative is finite.

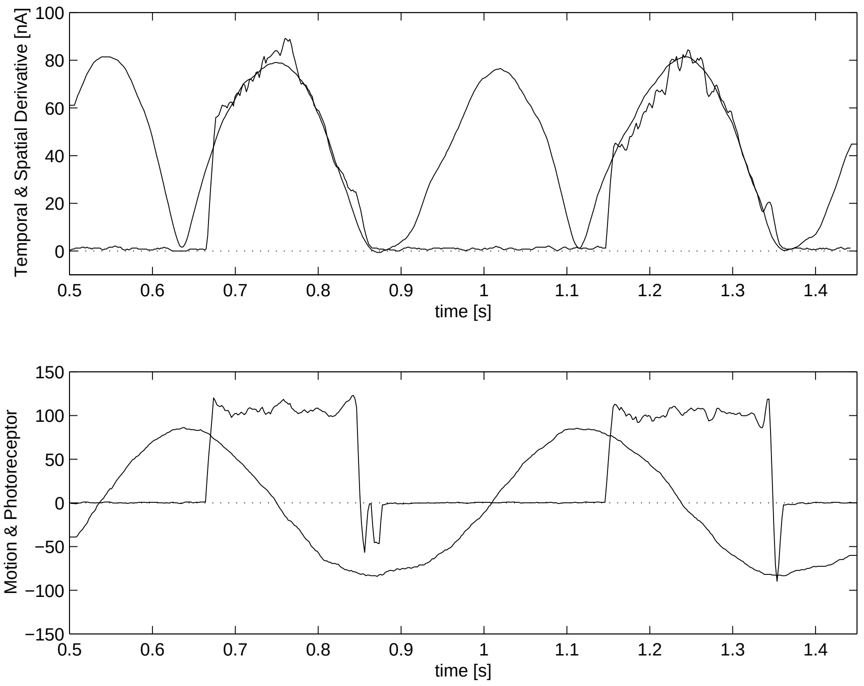


Figure 3.20: Solution to division by zero problem: A constant is subtracted off the temporal derivative. The height of the velocity output is not perceivably affected.

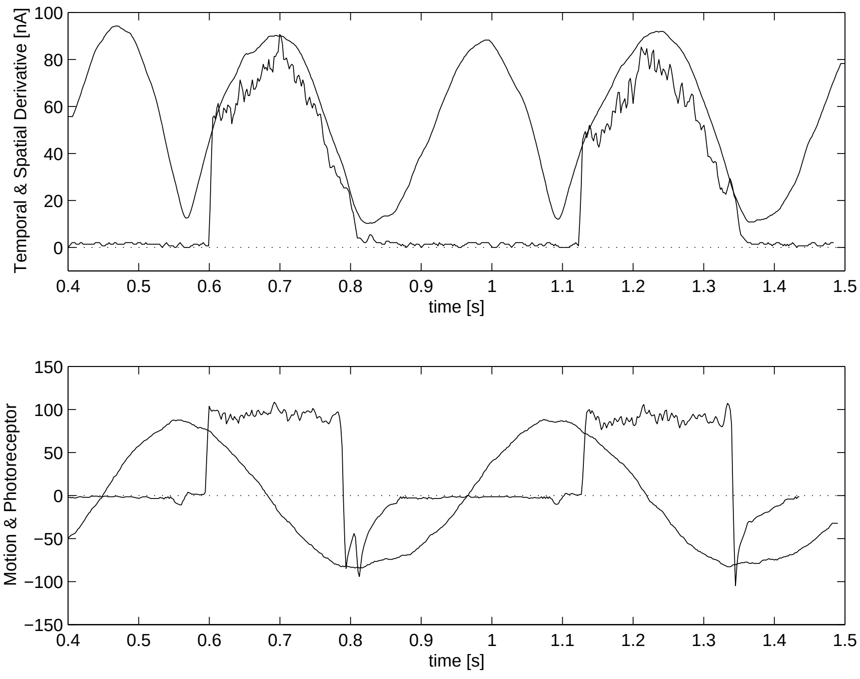


Figure 3.21: Solution to division by zero problem: A constant is added to the spatial derivative, its DC level is increased. Thus the heights of the velocity output is decreased.

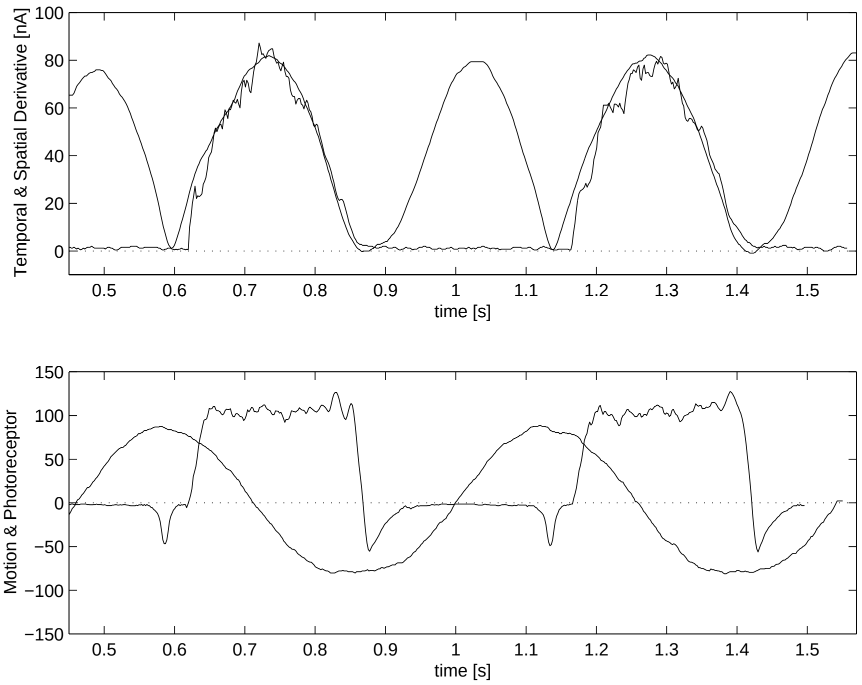


Figure 3.22: Solution to division by zero problem: The sign correction circuit can be slowed down to prevent fast changes in the sign. The edges of the velocity signal become shallower.

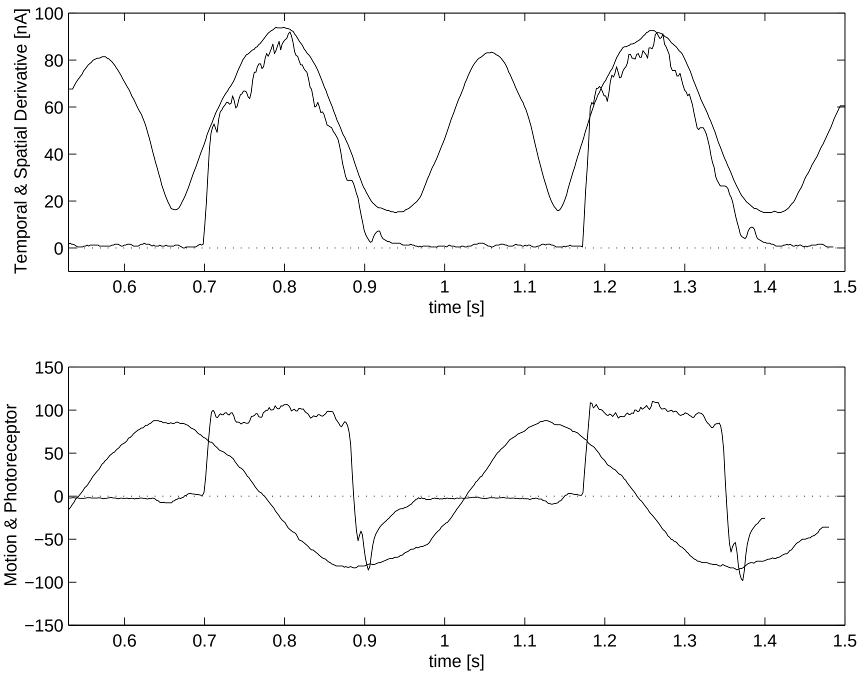


Figure 3.23: If the absolute value circuit is biased too high, then the spatial derivative becomes asymmetric and its DC level rises, the velocity output decreases.

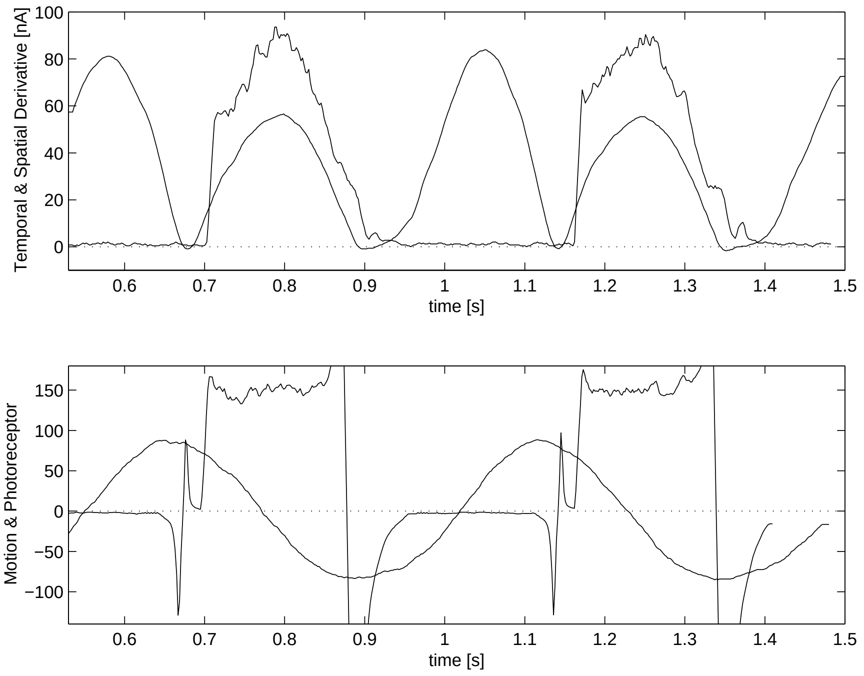


Figure 3.24: If the absolute value circuit is biased too low, the spatial derivative becomes asymmetric, too, causing an increased velocity output and division by zero problem.

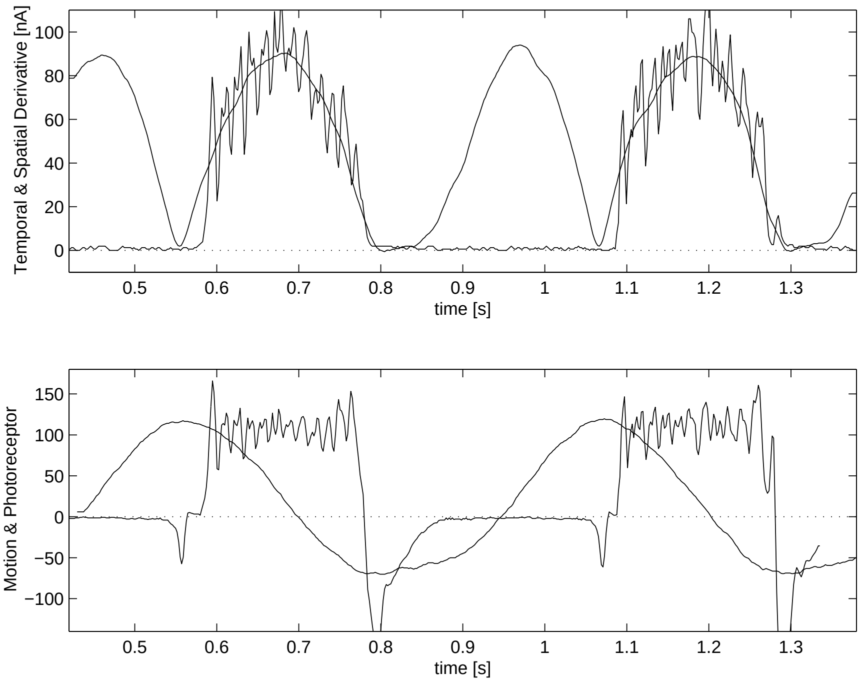


Figure 3.25: A strong photoreceptor bias causes oscillations in the temporal derivative and velocity output. The mean of the outputs, though, is still correct.

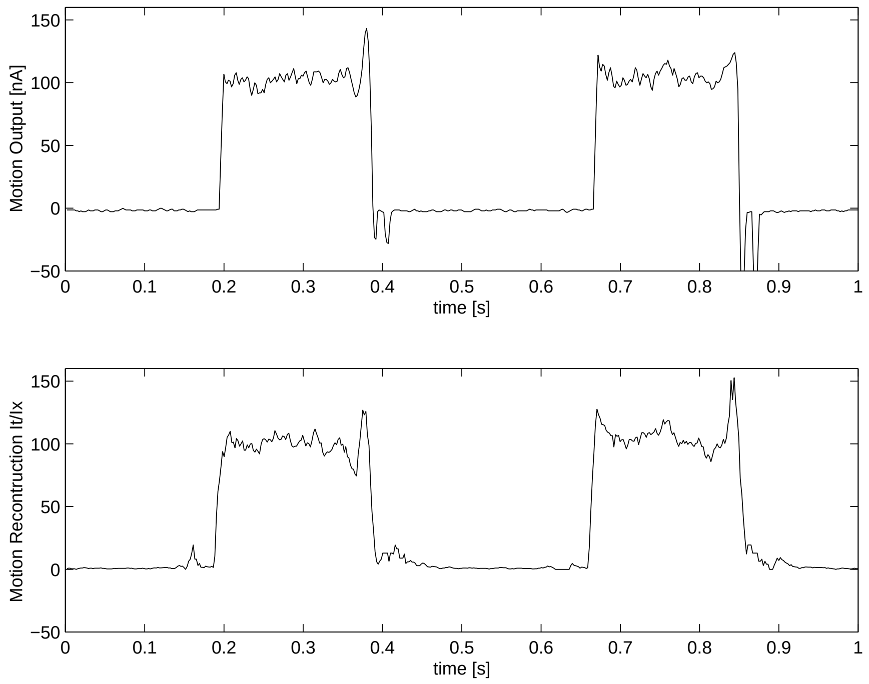


Figure 3.26: Sample: Reconstructed velocity output. The measured velocity (top trace) is compared to the result of dividing the measured temporal derivative current by the measured spatial derivative current (lower trace). Both traces are similar.

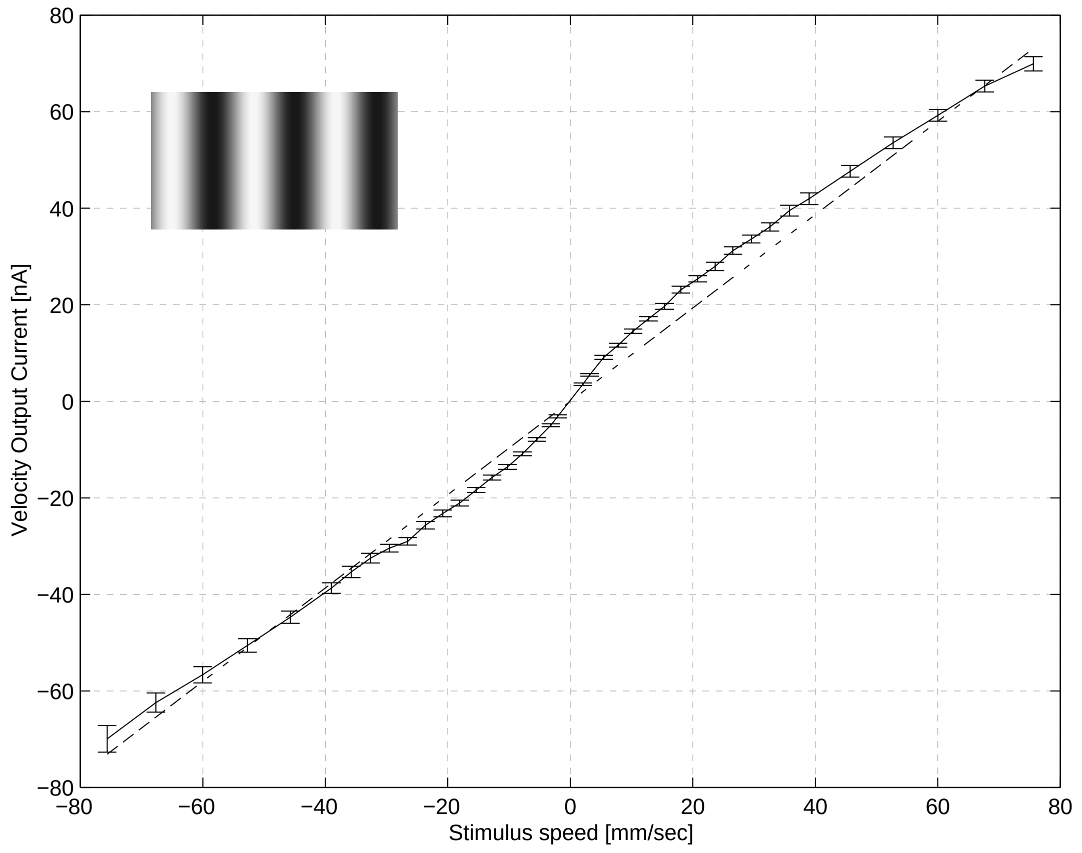


Figure 3.27: Velocity output of a single velocity computing unit of the CMotion1d sensor for stimulus velocities from 2 mm/sec up to 76 mm/sec in both directions. The stimulus was a sine wave of 72.5% contrast, as shown in the inset. The velocity output is plotted together with a linear fit. The linearity of the output is high, the nonlinear contributions are explained in the text.

3.6 Results: Velocity Dependence

In this section experiments are presented that show the CMotion1d sensor performance for stimuli of different velocity. As derived earlier the velocity output of the sensor should be linear with the stimulus velocity even for a discrete implementation of the spatial derivative. In Figure 3.27 the velocity output obtained from single velocity computing units is presented. A sine wave stimulus of 72.5 % contrast and spatial frequency 0.05 cycles/degree was used. On chip velocities that could be obtained with the 'can' stimulus setup range from 2 mm/sec up to 76 mm/sec. The output was recorded for both directions of motion.

The reported velocity is almost linear with stimulus velocity over the whole range. The error bars represent one standard deviation on each side and are obtained over multiple presentations of the same stimulus. The coefficient of variation for the velocity output ranges from 0.02 to 0.1 and is generally larger for negative velocities (see Figure 3.28).

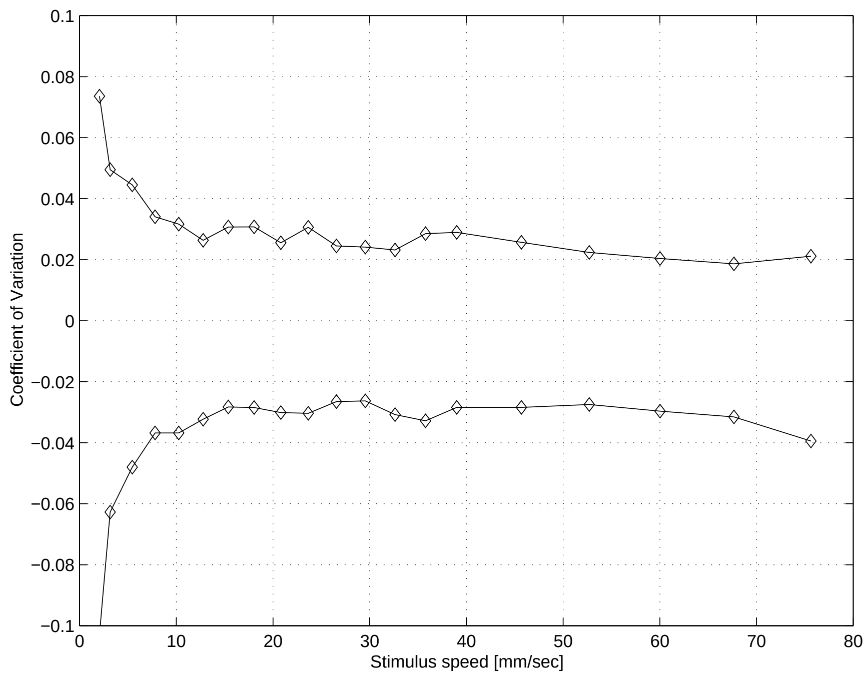


Figure 3.28: Coefficient of variation for velocity output. The fact that the cv for both directions are different indicates mismatch.

A constant gain offset was observed between both directions of motion. This gain offset has been compensated for in the presented velocity experiment by using slightly different bias voltages V_{mult} (see section 3.3) for both directions so that the dynamic range of sub-threshold currents could be used. The reason for the asymmetry between both directions of motion lies in the absolute value circuit.

In order to find out whether the velocity tuning of the CMotion1d was linear not only for pure sine wave stimuli but also for stimuli containing different frequency components, a saw tooth pattern was used. In Figure 3.29 the velocity output is demonstrated for this experiment. As can be seen not only the linearity is preserved but also the absolute magnitude is almost identical to the previous experiment where sine wave stimuli were used. This experiment shows that the CMotion1d sensor can be used also for stimuli of mixed frequency components.

Of great importance is the behavior of the CMotion1d sensor for stimuli of different contrasts. In a later section the velocity output will be shown for constant velocity but different contrasts. Here a sine wave of 45% contrast was used over the whole range of velocities. In Figure 3.30 the velocity output is shown. As expected the velocity response is not changed as compared to the high contrast stimuli. Due to the lower contrast the nonlinearity of the velocity response is slightly stronger. The absolute magnitude of the velocity output is larger because for this experiment a different gain bias V_{mult} was used. It will be shown later, though, that in fact also the absolute value of the velocity output for different contrasts is constant.

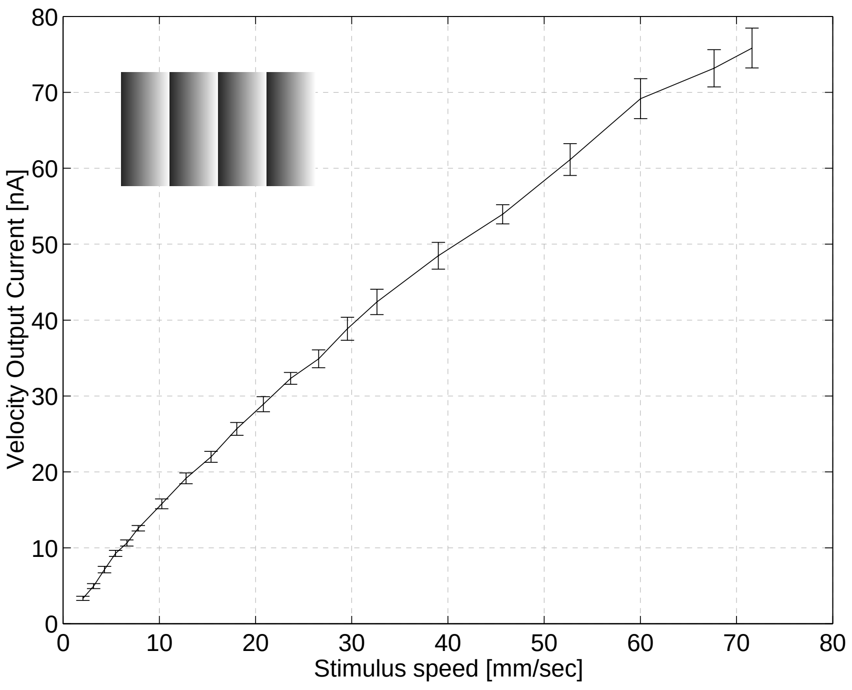


Figure 3.29: Velocity output for a single velocity computing unit for a saw tooth like stimulus (shown in the inset). Only one direction-of-motion is shown.

The velocities, over which the CMotion1d sensor has been shown so far to operate almost linearly, already covered most of the naturally useful range. It will now be shown, that the sensor operates correctly even for much lower stimulus speeds. For that purpose the 'bicycle wheel' device was used (see section 2.6.3) with saw tooth stimuli. Only one bias voltage was changed as compared to the fast velocity experiments: The temporal derivative bias V_{TD} was increased to amplify the temporal derivative signal. The results of this measurement are shown in Figure 3.31. It can be seen that the velocity output is almost linear over the entire range of stimulus velocities, that could be generated. This range spans from 0.073 mm/sec to 2.07 mm/sec on chip velocity. On the low end of the X axis the stimulus velocities were so slow, that no movement could be perceived with the eye. It also can be seen that the size of the error bars did not increase.

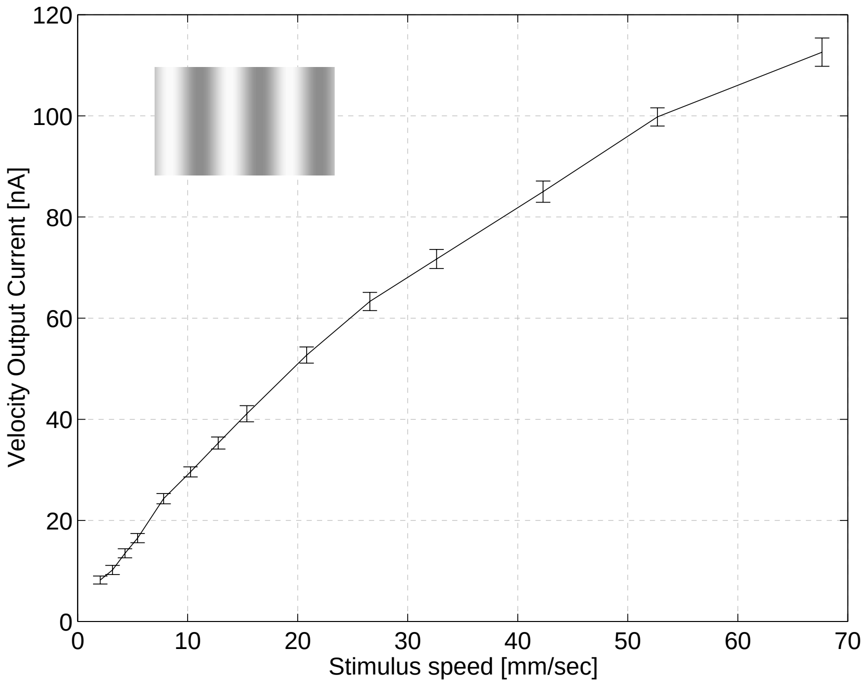


Figure 3.30: Velocity output for a sine wave stimulus of contrast 45% and spatial frequency 0.05 cycles/degree. Only one direction-of-motion is shown.

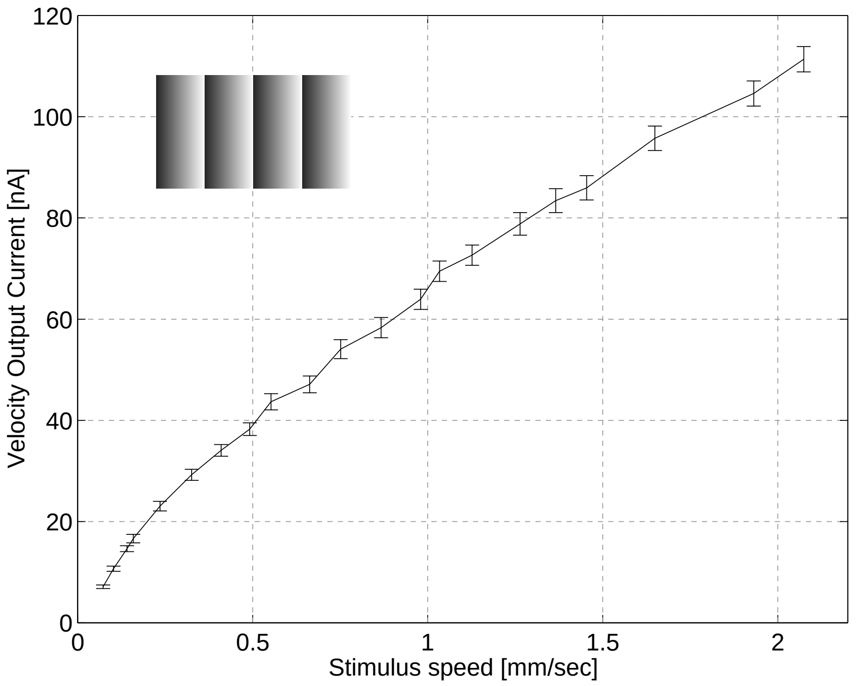


Figure 3.31: Velocity output for very low stimulus velocities. A saw tooth grating was moved at velocities as low as 0.073 mm/sec. The CMotion1d sensor was able to compute an almost linear velocity output even for this low velocities. Only the temporal derivative was amplified as compared to previous experiments to achieve measurable currents.

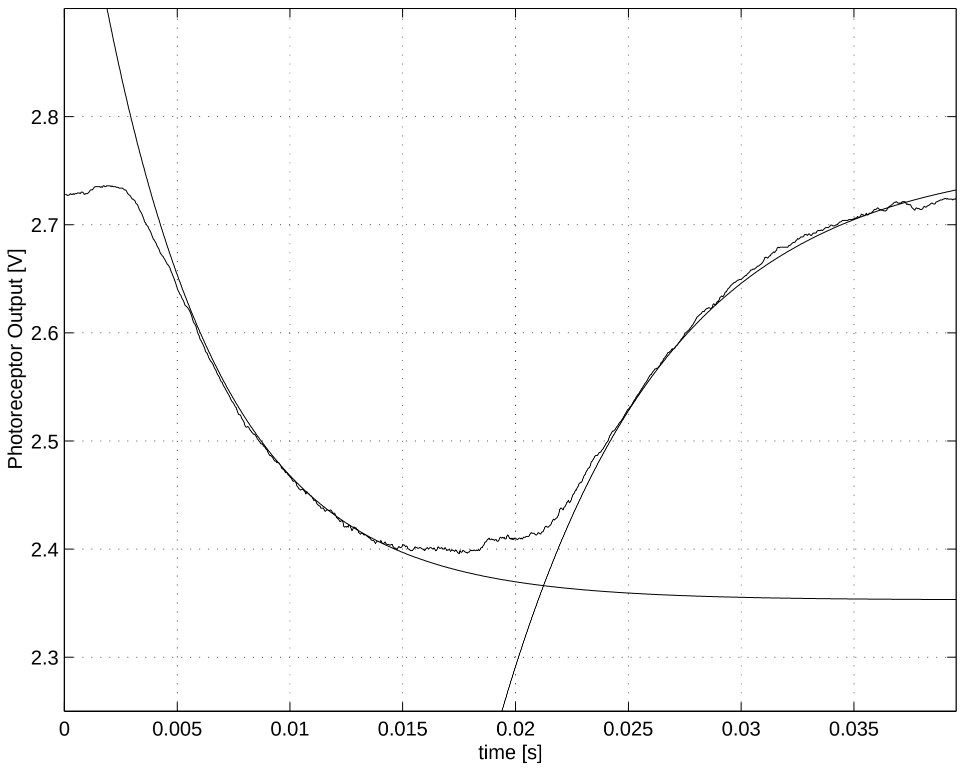


Figure 3.32: Step response of photoreceptor. A black stripe in front of white background was presented at high speed (77 mm/sec on chip velocity). Due to the low pass filter behavior the response of the receptor shows exponential decay and rise. The fit assumes a first order low pass filter model for the receptor. The time constant for the falling edge is 6 ms, corresponding to a cutoff frequency of 167 Hz. The time constant for the rising edge is 7.2 ms, corresponding to a cutoff frequency of 139 Hz.

Discussion of the velocity output

In this paragraph the nonlinearity of the velocity output will be discussed and explained. In the section about the fundamental limits already two possible contributions were postulated: Possibly mutually different time constants associated with the temporal and spatial derivative circuit and above threshold transistor operation. These contributions will be quantified with the measured data.

There are two ways to determine the first order temporal low pass filter behavior of the photoreceptor. First the transfer function of the receptor can be measured by presenting a sine wave stimulus of known spatial frequency and measuring the peak-to-peak amplitude of the receptor output for different velocities of the stimulus. The cutoff frequency can then be determined in a bode plot (see [Opp83]). Second an intensity step can be presented to the receptor. The output can then be fit with the step response of a first order low pass filter and the time constant can be determined. For determining the cutoff frequency of the receptor of the CMotion1d velocity sensor the second approach was chosen. Black stripes on white background were presented at high speed to the receptor. The high speed was necessary to make sure that the influence of the finite receptor size was negligible. In fact for slower velocities lower cutoff frequencies were measured due to the finite dwelling time of the edge on the active area of the photoreceptor. Additionally the lens has to be focussed such that blurring is minimized and the edge appears as crisp as possible on the chip.

Figure 3.32 shows the photoreceptor output for the stripe together with fits for the falling and rising edges. The respective fit functions were

$$V_{falling}(t) = V_1 - (V_1 - V_2)\,(1 - e^{-(t-t_{falling})/\tau_{falling}}) \tag{3.39}$$

$$V_{rising}(t) = V_2 + (V_1 - V_2)\,(1 - e^{-(t-t_{rising})/\tau_{rising}}) \tag{3.40}$$

where V_1 and V_2 are the upper and lower asymptotes, respectively. From the fit function $\tau_{falling}$=6 ms and τ_{rising}=7.2 ms are obtained, corresponding to cutoff frequencies $f_{falling} = 167$ Hz and $f_{rising} = 139$ Hz.

The fact that $\tau_{falling} \neq \tau_{rising}$ is intrinsic to the adaptive photoreceptor and shows that the receptor is not an ideal first order temporal low pass filter, where only one time constant is defined. The asymmetric temporal low pass filter behavior in the present implementation of the CMotion1d sensor has no effect on the velocity output. For the present CMotion1d implementation, only the falling edge is relevant.

The time constant τ_{TD} for the temporal derivative circuit was determined based on equation 3.34 and the time constant for the photoreceptor $\tau_{falling}$=6 ms. Figure 3.33 shows the peak value of the temporal derivative together with the best fit, which yields τ_{TD}=4 ms, corresponding to a cutoff frequency of f_{TD}=250 Hz. The fit is remarkably accurate.

The time constant τ_{SD} for the spatial derivative circuit was similarly determined with the fit function derived in equation 3.35. Figure 3.34 shows the peak value of the spatial derivative together with the best fit, which yields τ_{SD}=3.3 ms corresponding to a cutoff frequency of f_{SD}=300 Hz. The fit function including the two time constants approximates the data so well, that both data and fit cannot even be distinguished over most of the temporal frequency range. The data for photoreceptor and temporal and spatial derivatives was obtained in the same experiment as the velocity data shown in Figure 3.27. It is interesting to notice that the temporal derivative circuit has a larger time constant associated than the spatial derivative circuit, but both of these circuits operate much faster than the photoreceptor. In summary:

$$\tau_{falling} = 6\,ms > \tau_{TD} = 4\,ms > \tau_{SD} = 3.3\,ms. \tag{3.41}$$

Up to now the three time constants of the low pass filter model have been determined. These time constants can now be used in the equation for the expected velocity output (see equation 3.36) to find out how much of the non linearity of the measured velocity output can be explained by the low pass filter model. By doing so is turns out that the difference in τ_{TD} and τ_{SD} is too small to produce even a perceivable nonlinearity according to equation 3.36. This very important finding means, that time constants associated with computing circuitry in the CMotion1d sensor do not

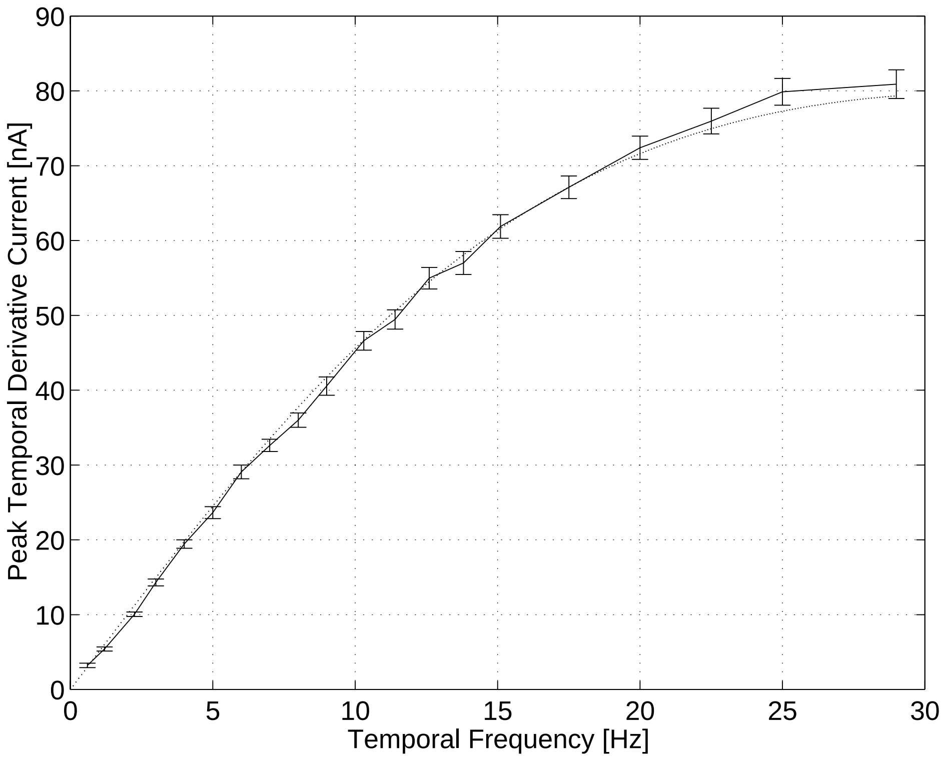


Figure 3.33: Velocity dependence of temporal derivative. The nonlinearity is due to temporal low pass filtering. From the best fit (dotted line) a time constant τ_{TD}=4 ms is obtained. On the abscissa instead of the stimulus velocity the corresponding temporal frequency $f = v_{stimulus}\, k/2\,\pi$ is plotted, where $k = 0.05\, cycles/deg$ is the spatial frequency of the stimulus.

affect the velocity output. The major contribution to the nonlinearity can be attributed to above threshold operation in the division circuit.

A small DC offset is observed in the velocity output for velocities other than zero, see for example Figure 3.27. The reason for this could be charge inequality on the floating gates of the division circuit, i.e. the chip was not exposed long enough the UV light.

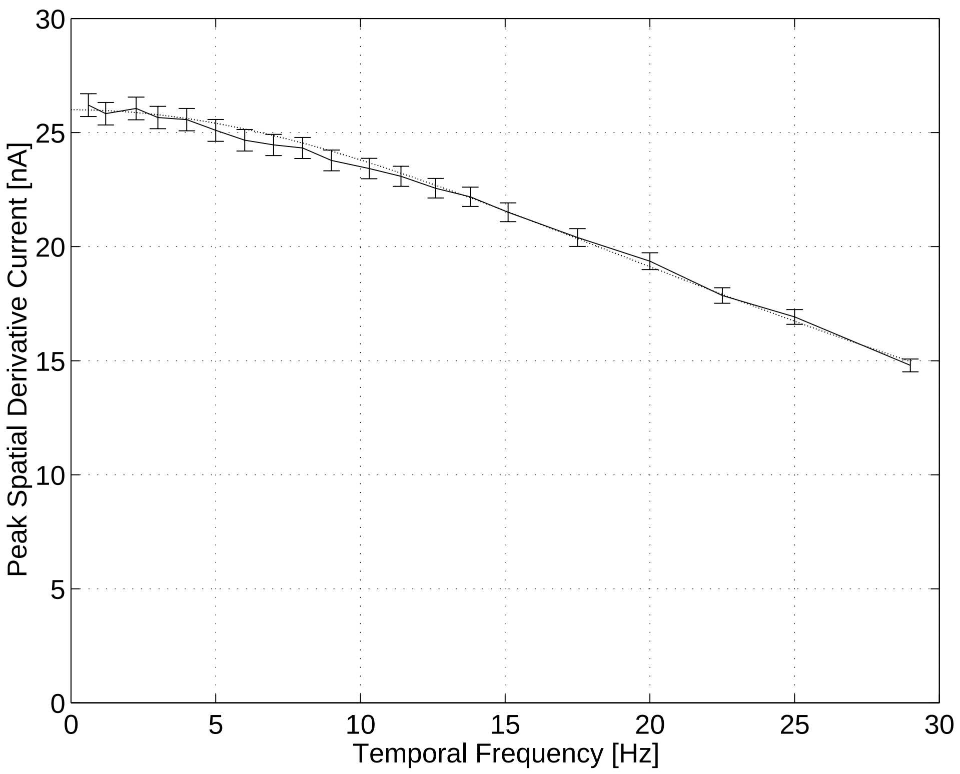


Figure 3.34: Velocity dependence of spatial derivative. Without temporal low pass filtering in the photoreceptor and the spatial derivative circuit a horizontal line is expected. The observed nonlinearity is due to temporal low pass filtering. From the best fit (dotted line) a time constant τ_{SD}=3.3 ms is obtained.

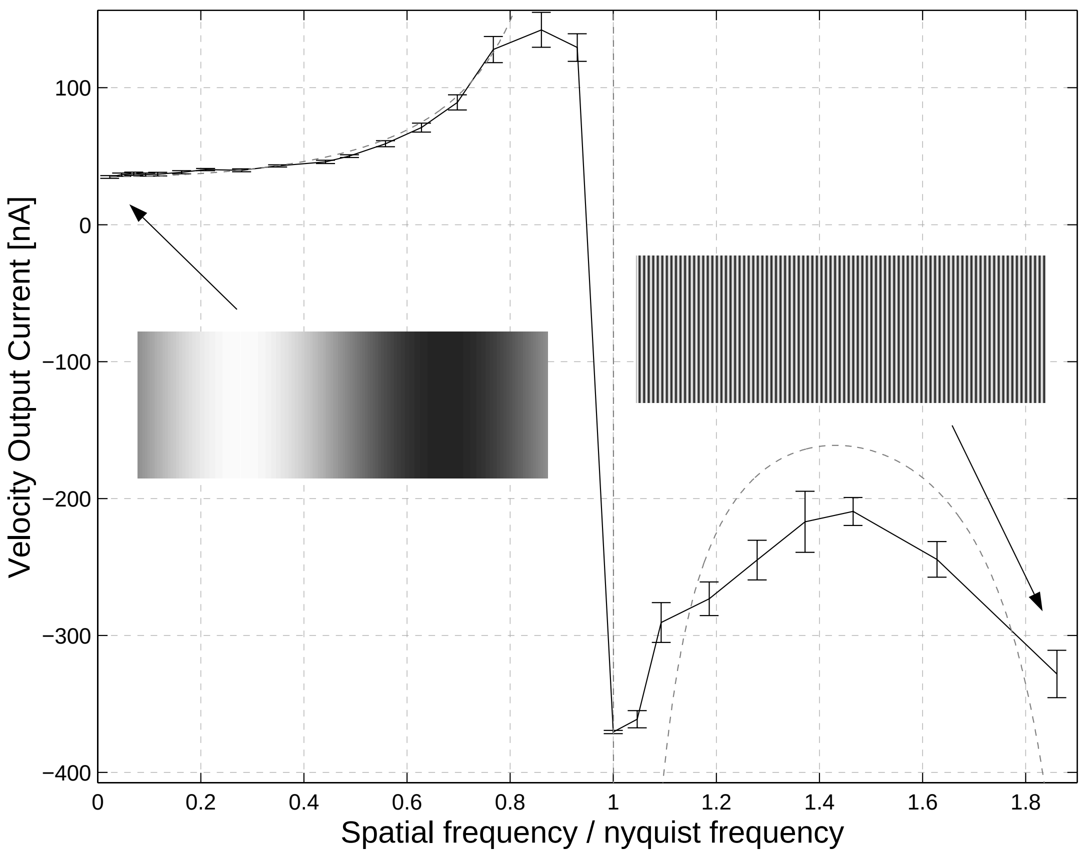


Figure 3.35: Spatial frequency dependence of velocity output. The velocity output is shown for stimuli of different spatial frequencies, the stimulus velocity was kept constant. The theoretical curve is plotted as dashed line. Two orders of magnitude of spatial frequencies were used. In the inset a low frequency and a high frequency stimulus is demonstrated. The velocity output matches the theoretical curve very well for spatial frequencies below the Nyquist frequency, and shows the correct shape even above the spatial aliasing point, where motion reversal occurs.

3.7 Results: Spatial Frequency Dependence

The ideal velocity sensor would provide a velocity signal which was independent of the spatial frequency of the stimulus. Mathematically the gradient method can provide this output. As derived in the fundamental limits section 3.4 any implementation of the gradient method, though, by the discrete nature of the spatial derivative must show spatial frequency dependence of the velocity output.

For the experiments of this section sine wave stimuli of different spatial frequencies were presented to the CMotion1d sensor while the velocity was kept constant.

Figure 3.35 shows that the sensor output meets the theoretical expectation for the spatial frequency dependence of the velocity output derived in equation 3.30 even beyond the aliasing point where the output reverses sign. The range of spatial frequencies used for this experiment is tremendous. Spatial frequencies from 0.01 cycles/degree up to 1 cycle/degree were presented to the sensor,

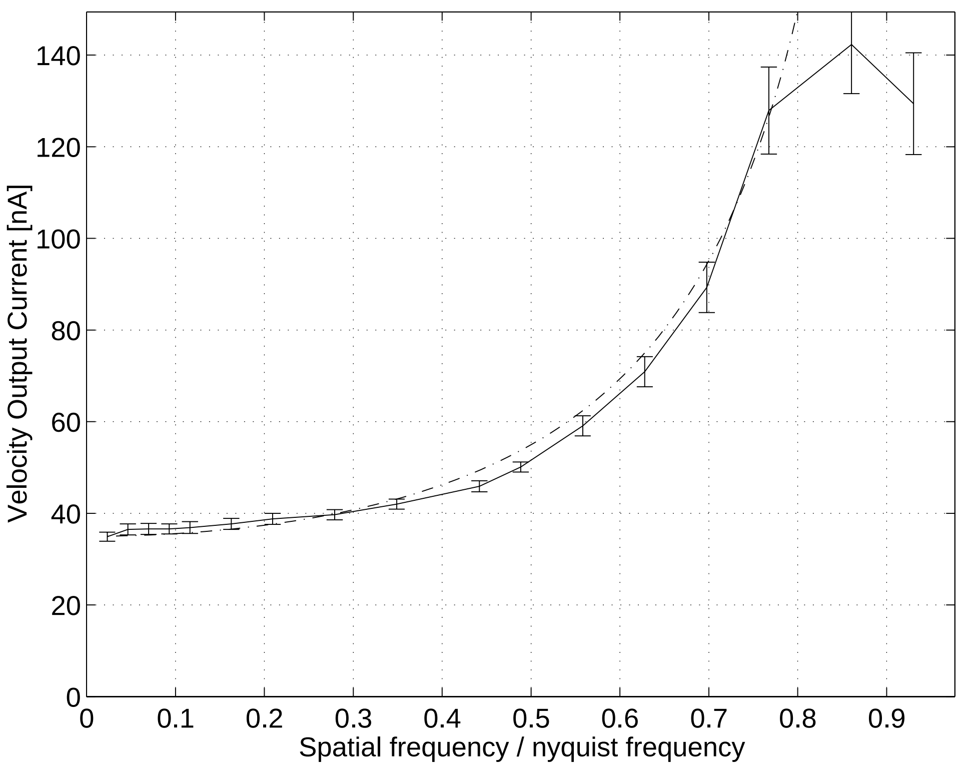


Figure 3.36: Spatial frequency dependence of velocity output. Shown is a magnification of the previous figure for spatial frequencies below the spatial aliasing point. As can be seen the theoretical curve, which is *not* a fit but obtained from the photoreceptor spacing, is met very well.

which represents a range of two orders of magnitude. In order to obtain a velocity output for high spatial frequencies the lens has to be focussed very precisely.

Figure 3.36 shows the lower frequency part of figure 3.35. It can be seen how well the theoretical expectation, plotted as dashed line, is met. For frequencies close the the Nyquist frequency the velocity output saturated due to above threshold operation of transistor devices. The dashed curve is *not* a fit, but rather a plot of the function $v\frac{\pi\, n}{\sin(\pi\, n)}$ where v is the constant velocity, $n = k/k_{nyquist}$, k the spatial frequency and $k_{nyquist}$ the Nyquist frequency. The Nyquist frequency $k_{nyquist}$ for a given implementation can easily be determined analytically. From the photoreceptor spacing Δ obtained from the layout and the focal length of the lens f_{lens}

$$k_{nyquist} = \frac{1}{\arctan 2\,\Delta/f_{lens}} = 0.42\, cycles/deg. \tag{3.42}$$

with $\Delta = 167\mu$m and f_{lens}=8 mm.

The velocity output due to the discrete nature of the spatial derivative is dependent on spatial frequency as predicted in the fundamental limits section 3.4. For a velocity sensor this is an undesired feature. Several ways exist to lessen this effect:

- Lens blurring can be used to filter out high spatial frequencies. If the spatial frequency compo-

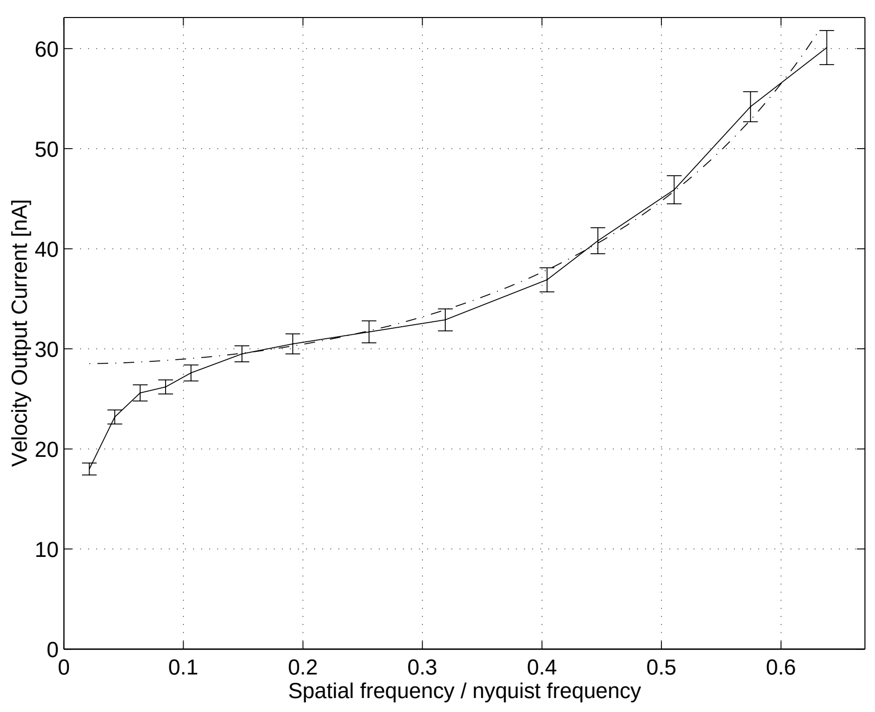


Figure 3.37: Spatial frequency dependence of the velocity output in the presence of a too strong leakage current $c_\triangle$. For small spatial derivatives this constant becomes significant and causes a too small velocity output.

nents of the input signals are kept below 0.3 times the Nyquist frequency, then only deviations smaller than 5% are obtained. For spatial frequencies $< 0.5\, k_{nyquist}$ deviations of 15% are to be expected.

- The smaller the pixel spacing, the larger the Nyquist frequency will be. A layout as tight as possible is thus desirable. In processes with higher integration density smaller pixel spacings can be achieved.

- For the experiments presented in this section a 8 mm lens was used. Choosing a longer focal length lens would make the field of view smaller and thus reduce the spatial frequencies of a given scene.

In Figure 3.37 the effect of a too strong leakage current $c_\triangle$ is measured. $c_\triangle$ is used to avoid the division by zero problem for spatial derivatives close to zero. But as demonstrated here, it can cause an unwanted decrease of the velocity output for low spatial derivatives. The current $c_\triangle$ can be set through the bias voltage V_{SDleak} such that this effect can almost be avoided.

In Figure 3.38 the spatial frequency dependence of the spatial derivative is shown. The dashed line shows the theoretically expected curve as derived in equation 3.29 for a temporally low pass filtered input. The time constant was determined in a separate experiment from the bode plot of the photoreceptor response and came out as $\tau_{falling}$=15.9 ms corresponding to a cutoff frequency

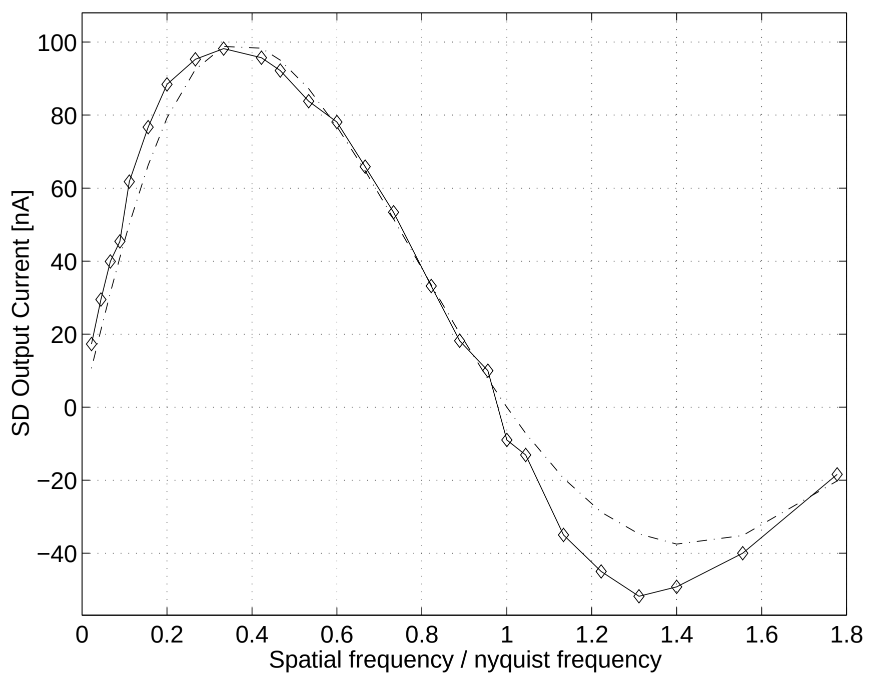


Figure 3.38: Spatial frequency dependence of spatial derivative. For a discrete spatial derivative a sine wave dependency is expected. The slight distortion is caused by temporal low pass filtering in the photoreceptor.

of $f_{falling}$=63 Hz. With this time constant the fit function was generated. As can be seen in this experiment the contribution of τ_{SD} is so minor compared to the slow photoreceptor time constant that is is not even required for the fit. Without any temporal low pass filtering for changing spatial derivatives a sinusoidal discrete spatial derivative is expected. The distortion is caused by temporal low pass filtering on the the photoreceptor.

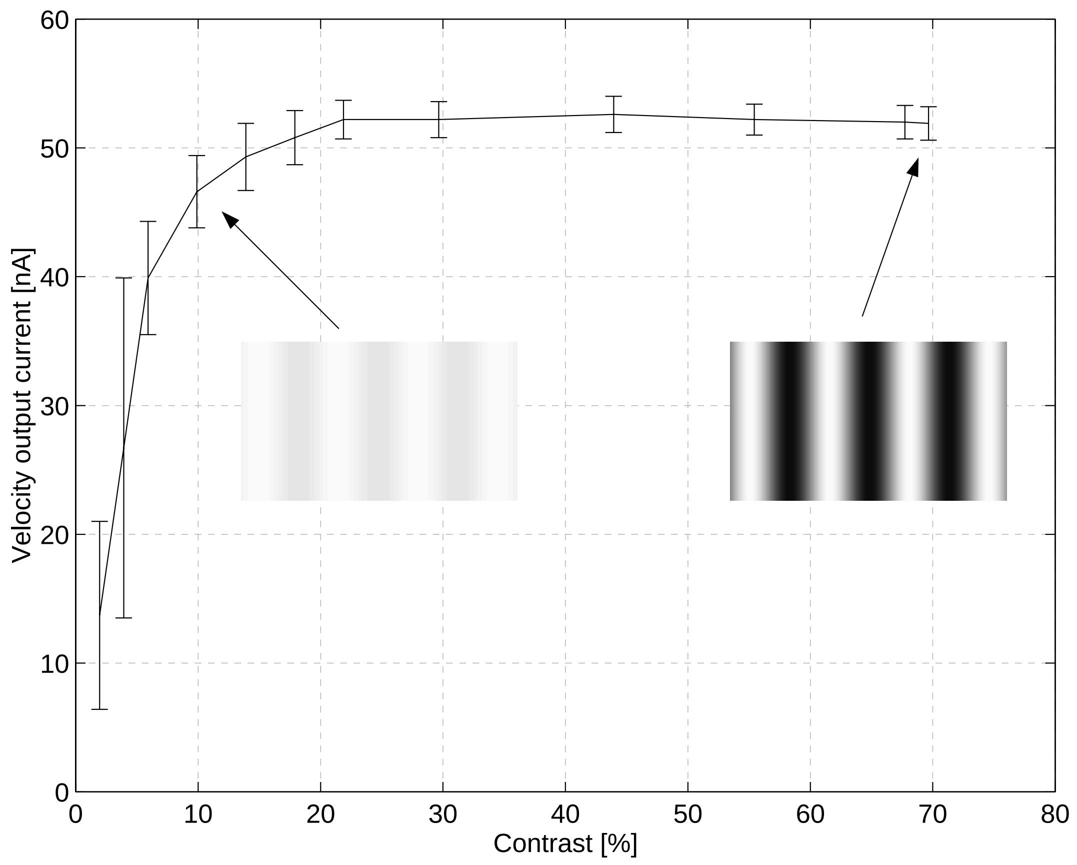


Figure 3.39: Contrast dependence of velocity output. The velocity output is shown for different contrasts. The stimulus velocity and spatial frequency was kept constant. In the inset a low contrast and a high contrast stimulus is demonstrated. The velocity output is independent of the stimulus contrasts for contrasts above 20 %. Even for contrasts as low as 4 % the correct direction-of-motion is computed.

3.8 Results: Contrast Dependence

In a multiplication based gradient scheme the velocity output is confounded with the contrast of the stimulus. This is because for constant velocity if the contrast increases also the spatial and temporal derivatives increase. Only by dividing those two magnitudes the contrast dependency cancels out.

Figure 3.39 shows the velocity output of the CMotion1d sensor for the whole range of printable contrasts from 2 % up to 70 %. The stimulus was a sine wave grating of spatial frequency k=0.05 cycles/deg which was kept at the constant velocity of v=10 mm/sec. It can be seen that the output is independent of contrast for contrasts above 20 %. The deviation over the range from 20 % to 70 % contrast is less than 1 %. Even for contrasts as low as 10 % the deviation of the output does not exceed 10 %. For very low contrasts the output goes to zero. This is a very important feature which has been added to the sensor: A small constant c_t is constantly subtracted from the temporal

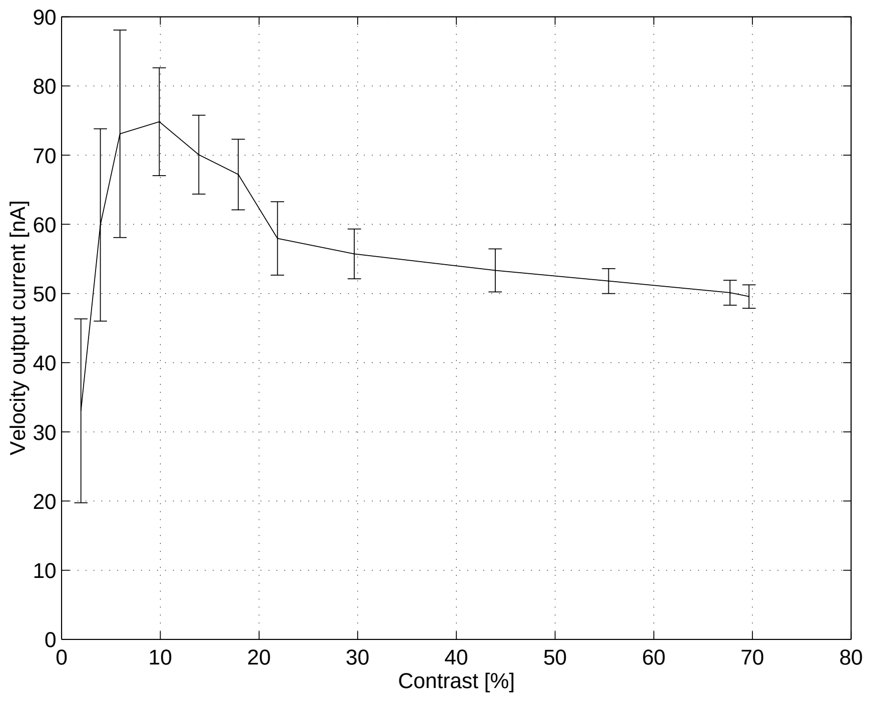


Figure 3.40: Contrast dependence of velocity output for a wrongly adjusted absolute value circuit. For decreasing contrasts the spatial derivative approaches zero faster than the temporal derivative, thus the velocity output increases. Eventually for very low temporal derivatives the velocity output goes to zero.

derivative and a different small constant $c_{\triangle}$ is constantly added to the spatial derivative such that the sensor output in fact is given by equation 3.37.

Figure 3.40 shows the effect if the absolute value circuit is not properly adjusted. If the bias voltage V_{abs} is wrong then for decreasing contrasts the spatial derivative approaches zero faster than the temporal derivative and the velocity output increases. Eventually for very low temporal derivatives the division circuit will always yield zero output.

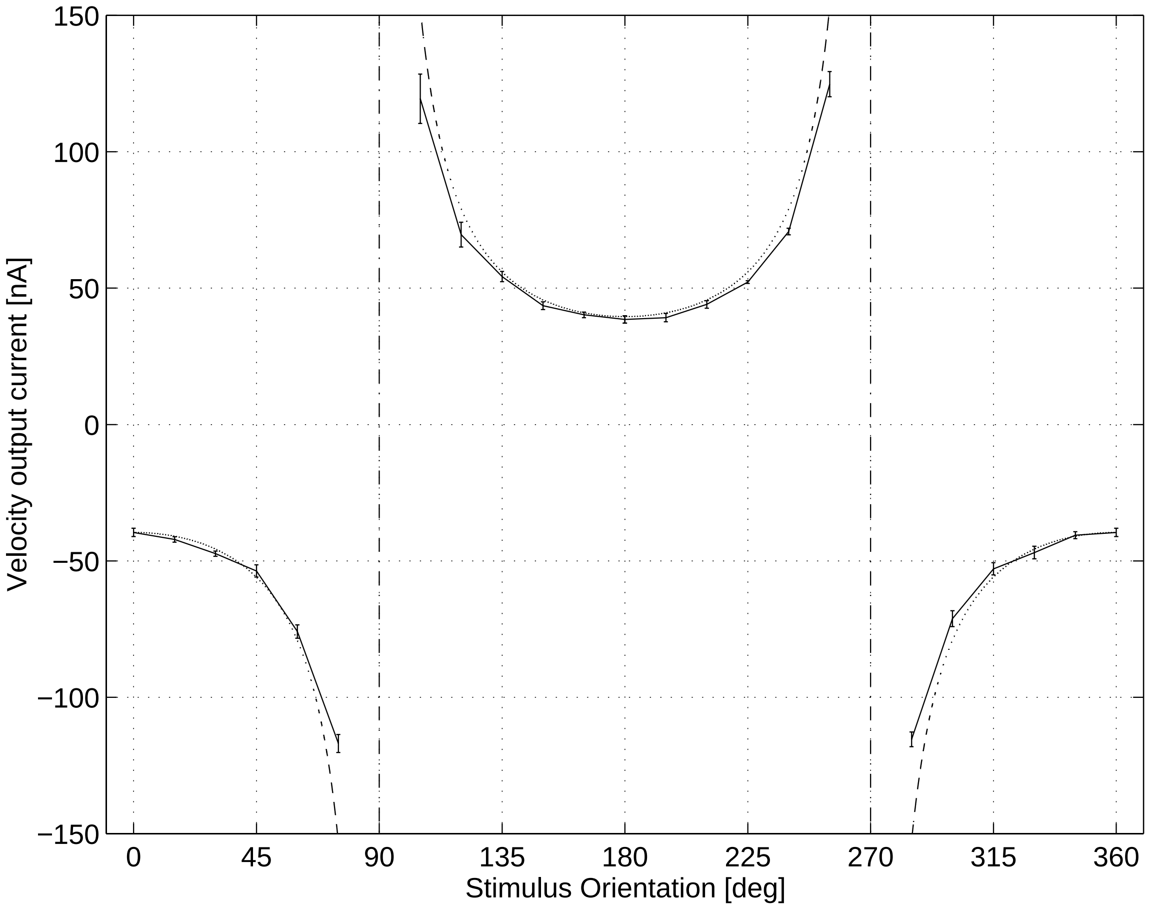


Figure 3.41: Orientation tuning curve of the CMotion1d sensor for a sine wave grating oriented from 0 to 360^o. The dashed line represents the theoretical expectation of a $1/\cos(\varphi)$ function, where φ is the stimulus angle.

3.9 Results: Orientation Tuning

An orientation tuning curve for a 1-D sensor is not strictly defined. For a 1-D sensor only stimulus movements within the sensor dimension are allowed. In practise, though, even if the sensor is only designed to measure velocity in one dimension, stimuli can be oriented arbitrarily. In Figure 3.41 therefore the measured orientation tuning curve of the CMotion1d sensor is shown together with the theoretical expectation as derived in equation 3.38. The stimulus was a sine wave grating of spatial frequency 0.05 cycles/deg and the constant velocity was 5.5 mm/sec. The expected $1/\cos(\varphi)$ behavior is followed very accurately. The sensor output in the orientation range from 90^o to 270^o was multiplied by a factor 1.9 in order to compensate for the gain difference between leftward and rightward motion. Except for this constant gain factor the outputs for both opposing directions are very similar. This is demonstrated in Figure 3.42.

In Figure 3.43 the same experimental data is shown in a polar plot. For comparison a $1/\cos(\varphi)$ curve is plotted, which is a straight line in a polar representation.

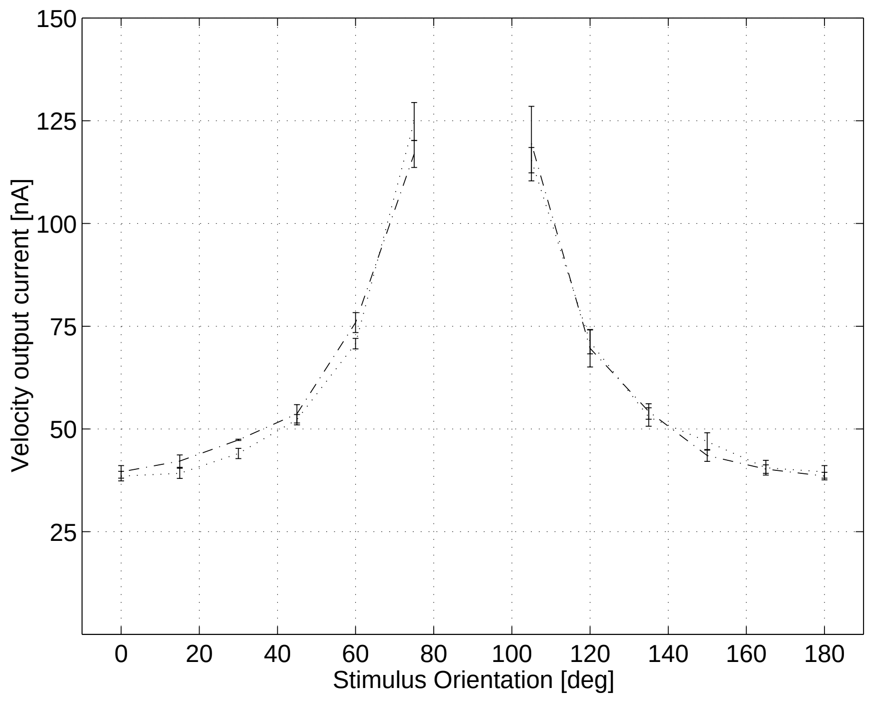


Figure 3.42: Absolute value of the velocity output for opposing stimulus orientations (dashed and dotted lines). Except for a constant gain difference between opposing directions, which has been compensated for, the output in both directions is similar.

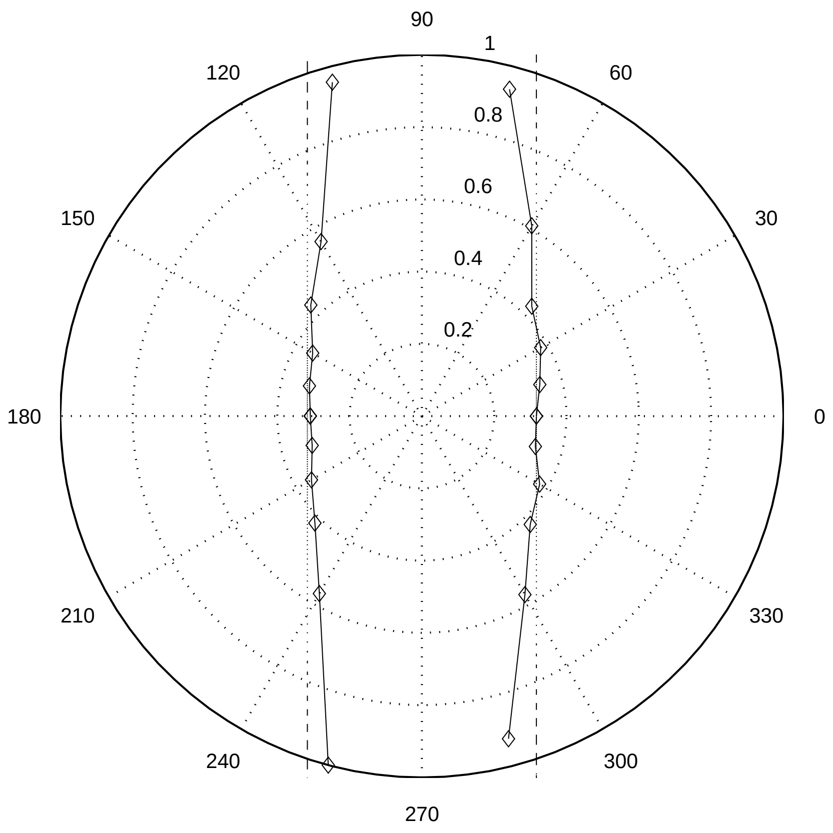


Figure 3.43: Polar plot for orientation tuning. This plot shows how close the orientation tuning comes to the $1/\cos(\varphi)$ behavior which is theoretically expected. It also becomes obvious, though, that the sensor must not be used for stimulus orientations other than 0^o and 180^o, because due to the orientation tuning the velocity output will not report the correct stimulus velocity in this case.

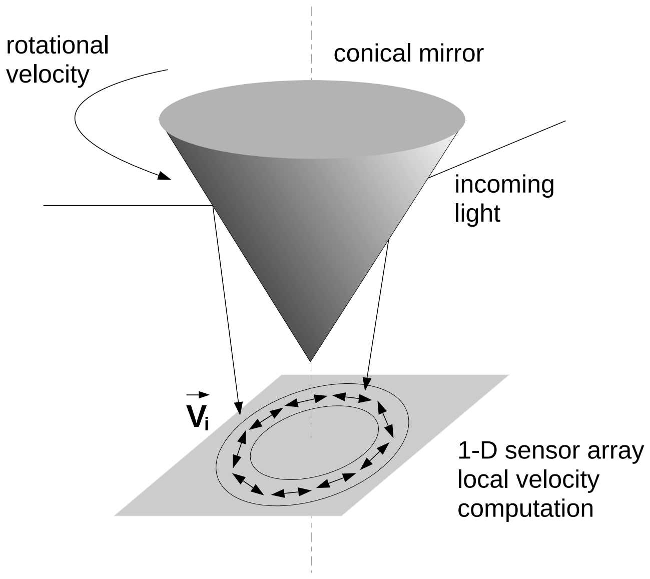


Figure 3.44: Setup for computing rotational velocity.

3.10 Computing Rotational Velocity

There are applications where ego-motion is of interest. For example in robotics ego-motion information is used for recovery of location in space. For that purpose mobile robots often rely on the velocity signal from their wheels, which is error prone due to slipping and sliding. Purely visual velocity determination has advantages because no moving mechanical parts are involved. Visual velocity determination in conventional machine vision systems usually involves a CCD camera and a μProcessor, with power consumption that often forbids long term stand alone applications. In this section a compact low power sensor system is proposed for real time computation of ego-rotational velocity. The system consists of a custom designed layout of 1-D velocity sensors and a conical mirror which reflects the intensity distribution in the horizontal plane onto the chip (see Figure 3.44). The layout is such that the elementary velocity computing units are arranged in a circle with equal inter pixel spacing. This type of custom layout as compared to the standard array structure is difficult to build because all pixels and scanners have to be arranged and connected up individually by hand. For this work the 1-D velocity computing unit of the CMotion1d velocity sensor was used.

In order to obtain the rotational velocity the signals of the local velocity computing units on the sensor are added together. This can be done with the type of scanner used for this work very conveniently by enabling all pixels at the same time through bits in the shift register (see section 2.6.1). To test the chip performance in the following section instead of using the conical mirror equivalently a stimulus rotating in the horizontal plane was presented to the sensor from above.

Figure 3.45 shows the sensor architecture on the chip level. The actual layout shown in Figure 3.11 can be recognized. Twenty full pixels, which include the velocity computing circuitry were arranged in an approximated circle. Eight photoreceptors were added in the corners of the chip to provide the intensity information for the spatial derivative. Without this precaution the distance between the pixels would not be equal throughout the chip and the corner pixels would give a too small velocity output. The distances between the photo-active areas in the actual implementation range from 154μm to 203μm. The full pixels were connected to one scanner cell each. In one mode of operation through the scanner cells only one pixel is reporting its results off chip. If the chip were to be used in averaging mode, all scanner cells would be activated and thus the outputs of all pixels would be added together.

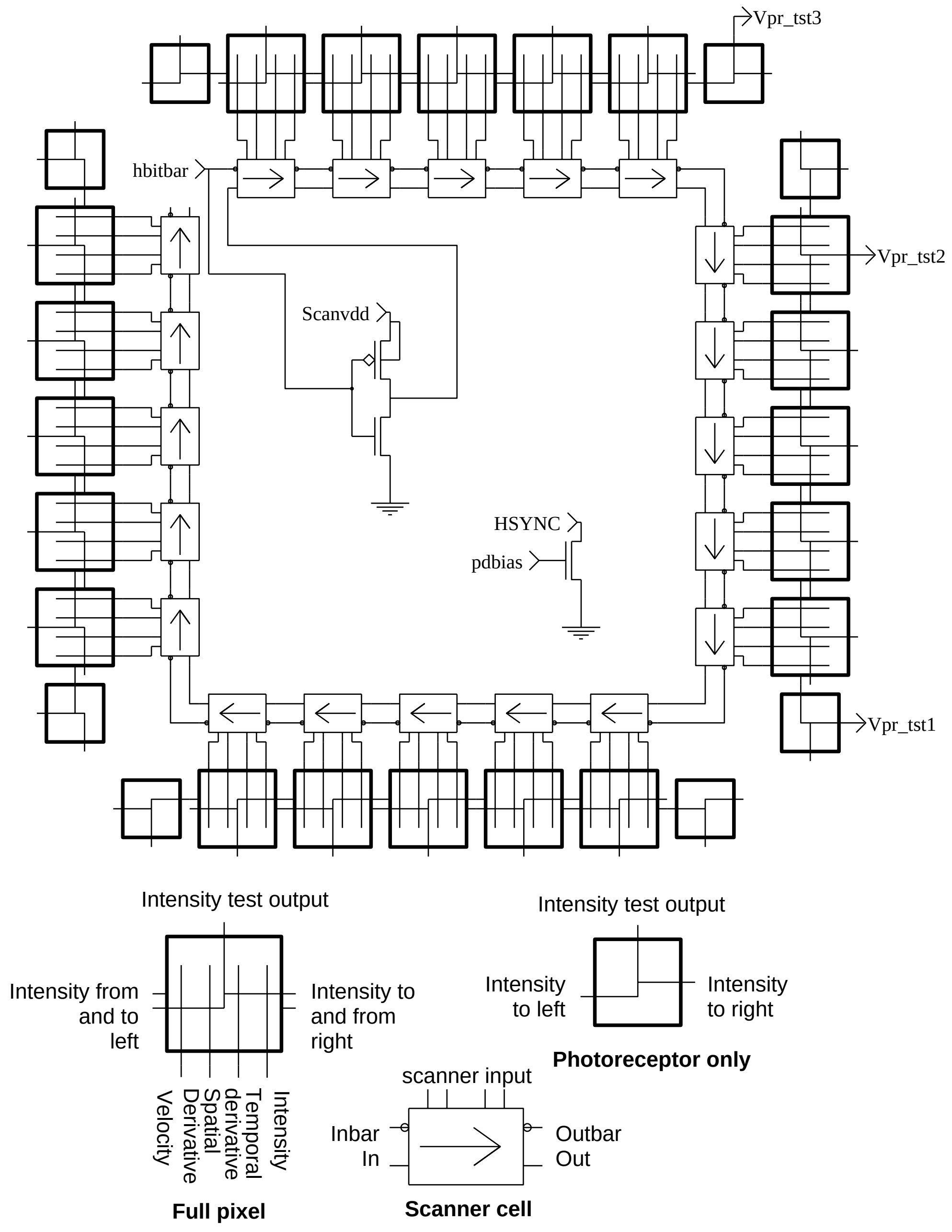


Figure 3.45: Chip schematic of the CMotion1d sensor. Main building blocks on the chip level are: The full pixels including the velocity computation, the photoreceptors used to keep the distances equal for the spatial derivatives and the scanner cells which are shift registers. The signal HBITBAR generates the bit for the first scanner cell, the signal HSYNC indicates if there is a bit in the scanner. Usually one pixel is giving its output on the global scanner output lines (not shown). In averaging mode the outputs of all pixels are summed together. Three of the photoreceptor test outputs are connected.

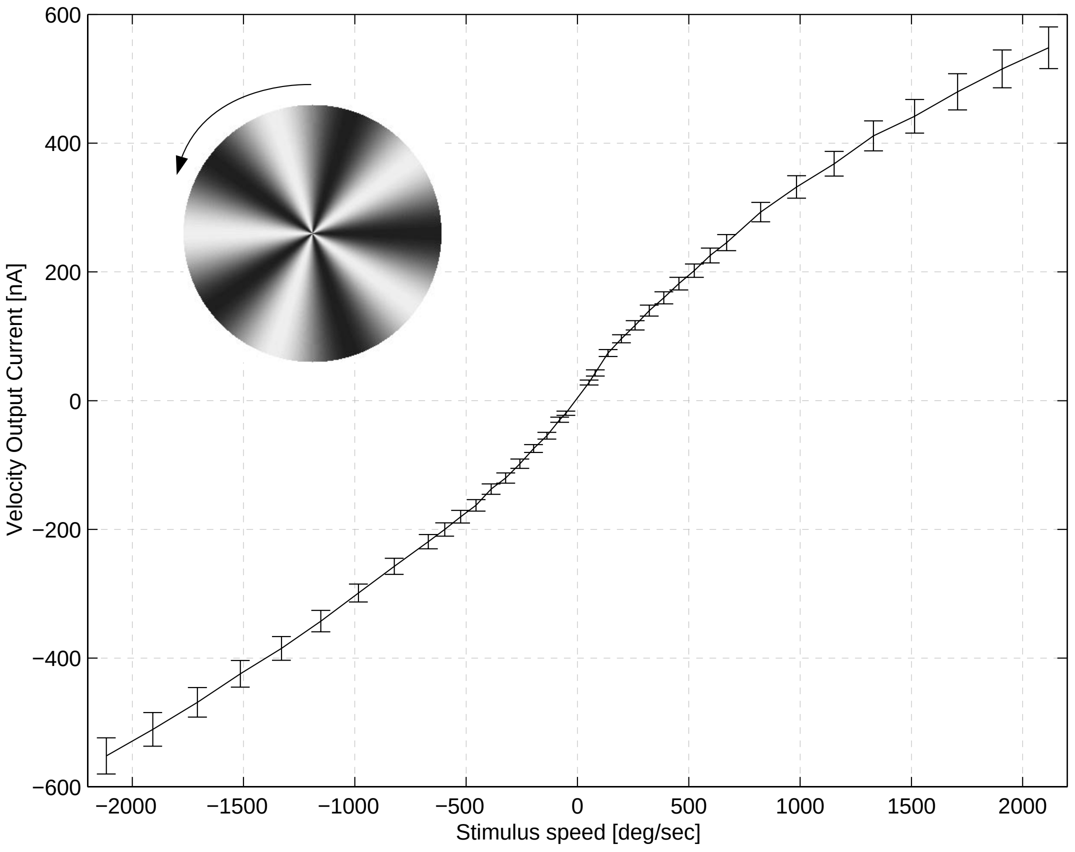


Figure 3.46: Rotational velocity output. The velocity of all 20 pixels was summed together while the stimulus depicted in the inset was rotated over the sensor. The error bars indicate the standard deviation after averaging over one period of the stimulus. The velocity output is almost linear with stimulus velocity up to 2120 deg/sec, which corresponds to 353 rpm. This range is very large as compared to what CCD based systems could bear.

3.11 Results: Rotational Velocity

For the experiment presented in this section all pixels were enabled through the scanner and a rotational sine wave grating was rotated above the sensor with axis of rotation at the position of the lens. Figure 3.46 shows the result for both left and rightward rotation of the stimulus. The velocity range over which the sensor operated almost linearly is remarkable. The velocities that could be obtained with the stimulus motor ranged from 50 deg/sec up to 2120 deg/sec. For computing the error bars the output of the sensor was averaged over one period of the stimulus. Without this averaging the error bars would be approximately four times larger. It can be argued that the largest contribution to the standard deviation stems from the fact that every single pixels generated a square step like output where the output is only valid during half of the time (see the next paragraph). Like in the single pixel output there was a gain difference between left and rightward motion, in this case the gain factor was 1.16. For the plot this gain offset has been compensated for.

In Figure 3.47 the summed temporal and spatial derivatives are shown. As in the case for single

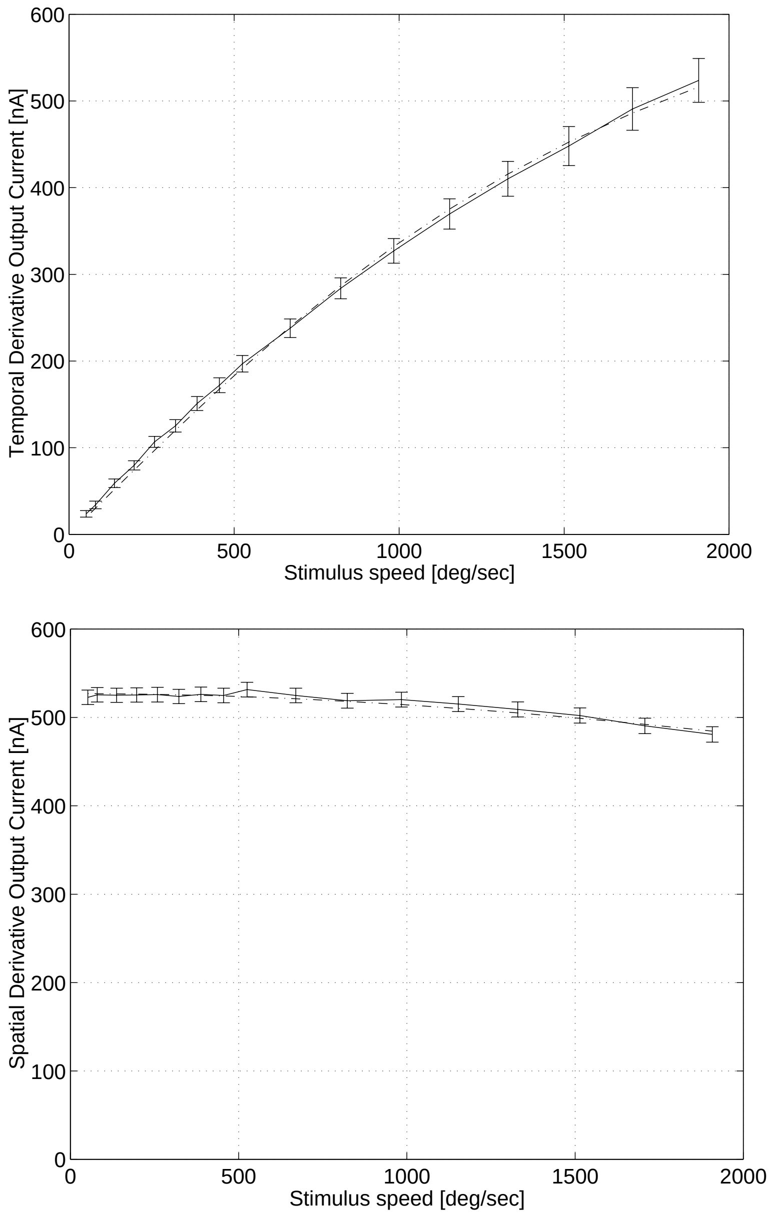


Figure 3.47: Summed temporal and spatial derivative. The measured derivative output is shown together with the theoretical expectation (dashed lines). As in the single pixel case, the slight non linearities can be explained by the low pass filter behavior of the photoreceptor. Filtering in the temporal and spatial derivative circuits plays a minor role.

pixels the low pass filter model approximates the non linearities very well. For this experiment the time constants that were found were again such that $\tau_{falling} > \tau_{TD} > \tau_{SD}$.

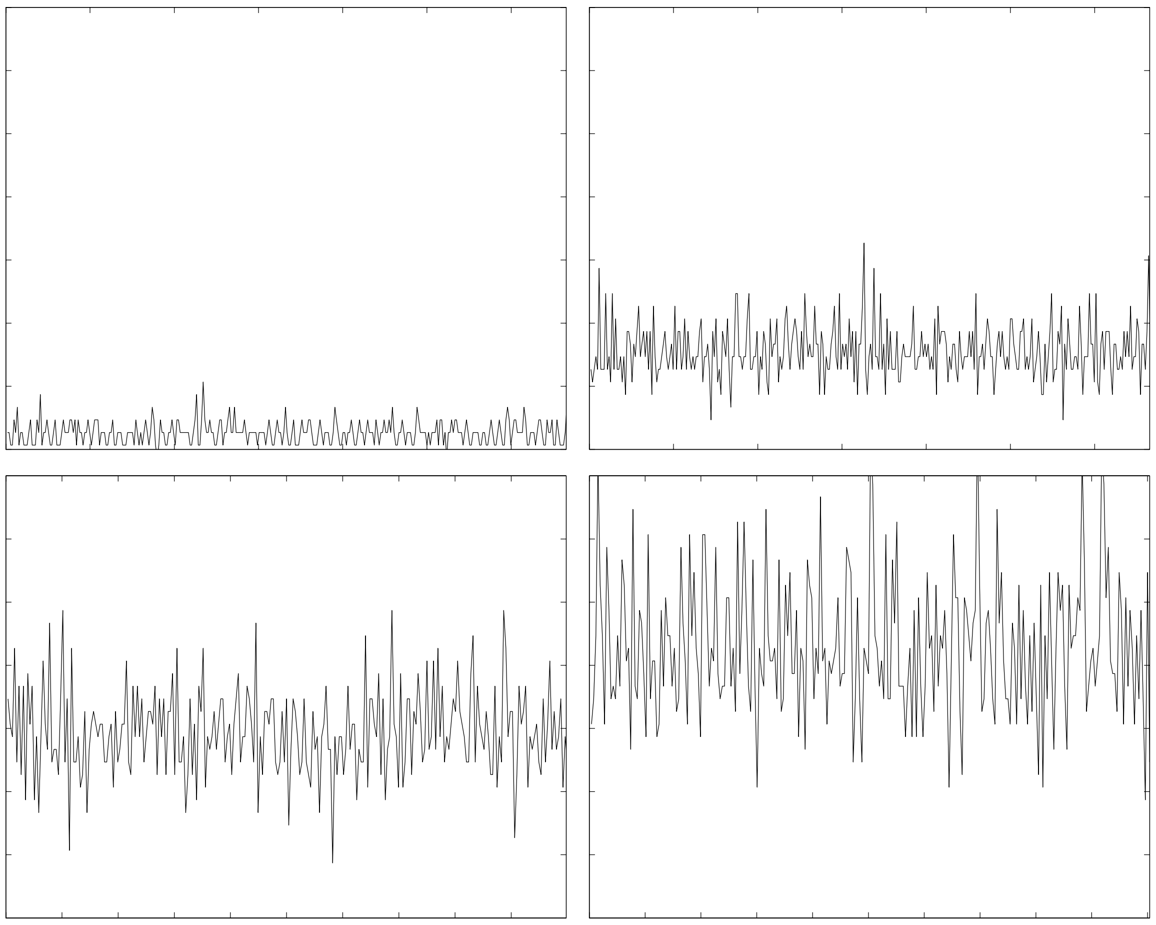


Figure 3.48: Summed velocity output of all 20 pixels over time. The Y-axis for all four plots ranges from 0 to 700nA. The output recording time is approx. 6 stimulus turns. The velocities were: 3 mm/sec, 21 mm/sec, 46 mm/sec and 76 mm/sec. As the stimulus velocity increases, the sensor velocity output and the standard deviation increase, too. The standard deviation is large because of the single pixel signal shape (see text).

Discussion of the standard deviation of the velocity measurements

The velocity signal obtained from one pixel of the CMotion1d sensor is valid during periods of decreasing intensity input. This is because only one sign of the temporal derivative is considered for the present implementation (see section 3.3 and Figure 3.49). For the experiments presented in this section a periodic stimulus is presented to the sensor and the output of all pixels is added together such that the mean velocity over the array is obtained. Due to the duty cycle of only τ/T for every single pixel there is an additional contribution to the standard deviation of the mean signal besides the other noise sources. In this section this standard deviation is derived. It will be shown that the error bars obtained for the averaged signal of all pixels are mainly due to the fact that the single pixel signals only have a duty cycle of $\tau/T = 1/2$. For a more detailed derivation see [Pap84].

The idealized signal (see Figure 3.49) obtained from one pixel is a square wave of height I_v and

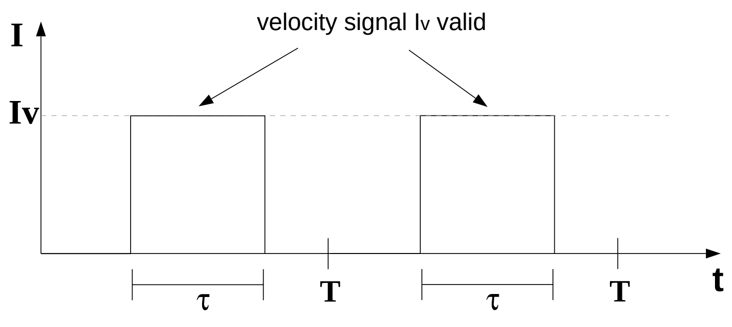


Figure 3.49: Idealized velocity signal obtained from one pixel with duty cycle of $1/2$. The velocity output is only valid for the times τ.

length τ. The signal is periodical with the period T. Therefore the duty cycle is τ/T.

The mean of the signal over time of one pixel is

$$\langle I\rangle_t = \frac{1}{T}\int_0^T I_v(t)dt = I_v \cdot \frac{\tau}{T}. \tag{3.43}$$

The variance of the signal of one pixel is given by

$$\begin{aligned}\sigma^2 &= \frac{1}{T}\int_0^T (I_v(t) - \langle I\rangle_t)^2 dt = \\ &= \frac{1}{T}\int_0^\tau (I_v - \frac{I_v\cdot\tau}{T})^2 dt + \frac{1}{T}\int_\tau^T (\frac{I_v\cdot\tau}{T})^2 dt = \\ &= I_v^2 \cdot \frac{\tau}{T}\cdot(1-\frac{\tau}{T})^2 + I_v^2\cdot\frac{T-\tau}{T}\cdot(\frac{\tau}{T})^2 = \\ &= I_v^2\cdot\frac{\tau}{T}\cdot(1-\frac{\tau}{T}).\end{aligned} \tag{3.44}$$

If all N pixels were completely uncorrelated, then the expected variance of the summed signal was:

$$\sigma^2_{uncorr} = N\cdot\sigma^2 = N\cdot I_v^2\cdot\frac{\tau}{T}\cdot(1-\frac{\tau}{T}) \tag{3.45}$$

whereas for completely correlated pixels we obtain

$$\sigma^2_{corr} = N^2\cdot\sigma^2 = N^2\cdot I_v^2\cdot\frac{\tau}{T}\cdot(1-\frac{\tau}{T}). \tag{3.46}$$

In the first case the coefficient of variation is then given by

$$cv_{uncorr} = \frac{\sigma_{uncorr}}{N\cdot\langle I\rangle_t} = \frac{1}{\sqrt{N}}\cdot\sqrt{\frac{T}{\tau}-1} \tag{3.47}$$

and in the second case for correlated pixels

$$cv_{corr} = \frac{\sigma_{corr}}{N\cdot\langle I\rangle_t} = \sqrt{\frac{T}{\tau}-1}. \tag{3.48}$$

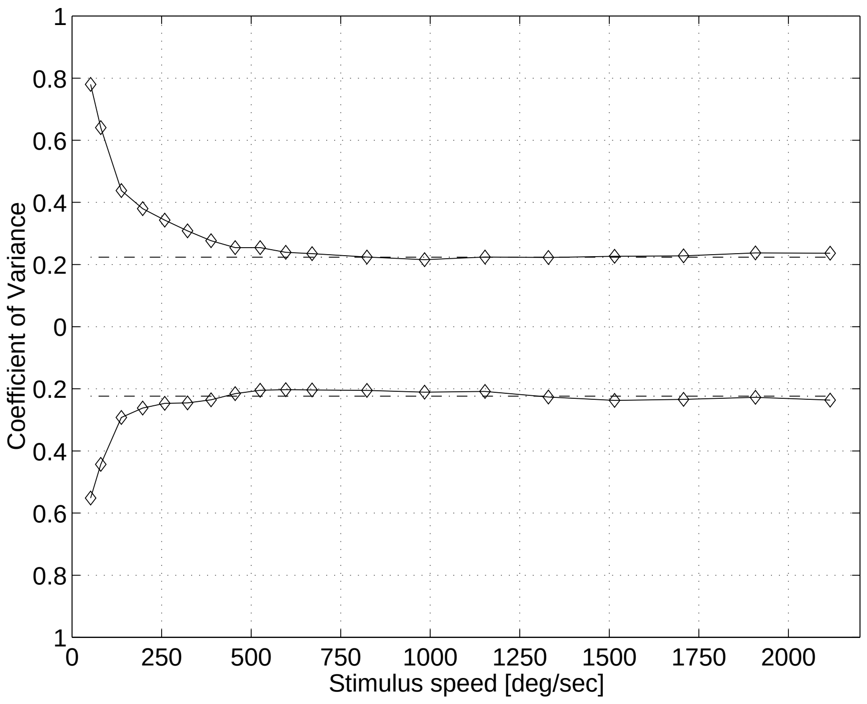


Figure 3.50: Coefficient of variance of the velocity output. For stimulus velocities above 500 deg/sec the cv value lies between 0.20 and 0.27. The expected cv for uncorrelated pixels is 0.22, and is drawn as dashed line.

If we assume a duty cycle of 1/2 then we obtain $cv_{uncorr} = 1/\sqrt{20} = 0.22$ and $cv_{corr} = 1$. The Coefficient of Variance for the experiment presented in this section is of the same magnitude as theoretically expected from the above considerations for uncorrelated pixels. For both directions of rotational velocity for velocities above 500 deg/sec the cv lies between 0.20 and 0.27 (see Figure 3.50). This means that the largest contribution to the measured standard deviation of the velocity signal stems from the fact that the single pixel signal has only a duty cycle of 1/2. We can assume that the signals obtained from the pixels were uncorrelated because due to the sine wave stimulus used in this experiment every pixels received an input intensity of a different phase.

3.12 Velocity2d: Gradient Method for 2-D Velocity

Encouraged by the fact, that the gradient method in 1-D was implementable in analog VLSI hardware, a 2-D gradient method sensor was developed. Due to the long chip fabrication time, results have not yet been obtained. Therefore in this section just the algorithm and implementation are outlined.

3.12.1 Algorithm

An equation will be derived which relates the change in image brightness at a point in a plane to the motion of the brightness pattern. A unique solution for the optical flow will be obtained with the additional constraint of normal flow.

Let the image brightness at the point (x, y) in the image plane at time t be denoted by $I(x, y, t)$. Now consider what happens when the pattern moves. The brightness of a particular point in the pattern is constant, so that

$$\frac{dI(x, y, t)}{dt} = 0. \tag{3.49}$$

Using the chain rule for differentiation

$$\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t} = 0 \tag{3.50}$$

a constraint on the velocity components $v_x = \partial x / \partial t$ and $v_y = \partial y / \partial t$ is obtained. An additional constraint is required for a unique solution for the local image velocity. Several constraints have been suggested in the literature, for example the smoothness of the optical flow field [HS81]. This approach is problematic because discontinuities in the optical flow field, on which segmentation from motion algorithms rely, are attenuated. A different constraint was used for the purpose of this work which maintains discontinuities and can be implemented in analog VLSI hardware. The optical flow is confined to be normal to the local image brightness distribution, i.e. it is oriented in the direction of the intensity gradient. In mathematical terms

$$\frac{I_x}{x_x} = \frac{I_y}{v_y} \tag{3.51}$$

where $I_x = \partial I / \partial x$ and $I_y = \partial I / \partial y$. By combining the two constraints 3.50 and 3.51 for the velocity it is found

$$\begin{aligned} v_x &= -\frac{I_x I_t}{I_x^2 + I_y^2} \\ v_y &= -\frac{I_y I_t}{I_x^2 + I_y^2} \end{aligned} \tag{3.52}$$

where $I_t = \partial I / \partial t$.

3.12.2 Implementation

The division of temporal and spatial derivative in the CMotion1d sensor implementation was already a fairly complicated task. For the purpose of 2-D velocity four more computational steps are required, as derived in the previous paragraph. Besides a division operation a multiplication, a summation and two squaring operations are required. These four operations have to be performed for both velocity vector components. The implementation of the gradient method in 2-D is further complicated by the fact, that only unsigned signals can be multiplied, squared and divided, whereas the derivatives in equation 3.52 as signed values.

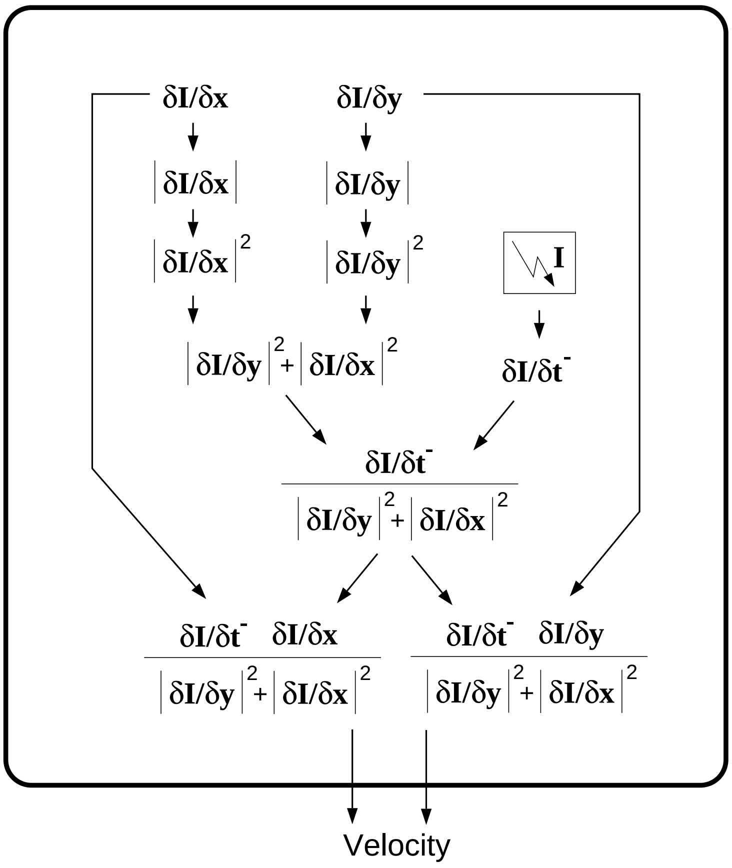

δI/δx
δI/δy
|δI/δx|
|δI/δy|
|δI/δx|²
|δI/δy|²
I
|δI/δy|² + |δI/δx|²
δI/δt⁻
δI/δt⁻ / (|δI/δy|² + |δI/δx|²)
δI/δt⁻ δI/δx / (|δI/δy|² + |δI/δx|²)
δI/δt⁻ δI/δy / (|δI/δy|² + |δI/δx|²)
Velocity

Figure 3.51: Block diagram of one Velocity2d sensor pixel. From the local adaptive photoreceptor output the temporal derivative is computed, only the negative sign is considered. The spatial derivatives in X and Y direction are computed from the photoreceptor values of the four neighbors in a wide linear range amplifier. The absolute values are taken and both spatial derivatives are squared in a νMOS squaring circuit. With a different νMOS circuit the temporal derivative is divided by the sum of the squared spatial derivatives. This value is finally multiplied with the signed values of the spatial derivatives to yield the velocity components in X and Y direction. Only two signals are passed between each two pixels: The photoreceptor voltage to and from the neighbor. From every pixel the light intensity, the temporal derivative, the absolute value of the spatial derivative in Y, the squares of the spatial derivatives in X and Y, the result $I_t/(I_x^2 + I_y^2)$ and the velocity signals in X and Y are obtained.

Interestingly besides the Tanner-Mead chip, which was built to compute only global velocity [TM86], the Velocity2d sensor outlined here is the first example of a velocity sensor that combines information of both spatial dimensions for both velocity components. To the knowledge of the author all other 2-D motion and velocity sensors reported in the literature are mere 1-D sensors that have been duplicated for both spatial dimensions.

The block diagram in Figure 3.51 shows how the gradient method in 2-D was implemented. As in the CMotion1d sensor the spatial derivative is computed from the light intensity of the neighbors in a novel wide linear range amplifier. Also like in the CMotion1d sensor the temporal derivative is computed from the local light intensity and only one sign is considered. The spatial derivatives are subsequently rectified, squared and summed. The temporal derivative is the divided by this sum, such that both velocity components can then be obtained by multiplication with the spatial derivative components. For testing purposes not only the final velocity result can be read off the sensor, but also the light intensity, the temporal derivative, the absolute value of the spatial derivative in Y, the squares of the spatial derivatives in X and Y, the result $I_t/(I_x^2 + I_y^2)$ and the velocity signals in X and Y can be obtained from every single pixel in the sensor array.

In Figure 3.52 the circuit diagram for one pixel in the Velocity2d sensor is shown. As can be seen, the implementation of the true gradient method for 2-D normal optical flow is very complex. On a tiny chip of the physical size 2.2 mm$\times$$2.2$ mm a 5×5 array was achieved. The pixel size was 292μm$\times 292\mu$m. In Figure 3.53 the layout of one pixel in the Velocity2d sensor is shown. It can be seen that not only the transistor count is high, but also a fair amount of chip real estate is required for infrastructure: Poly wires for the biases and scanner clock running horizontally and the metal bus wires for the readout running vertically. The size of the photodiode in the upper left corner is 32μm$\times 32\mu$m, thus the fill factor is only 1.2%.

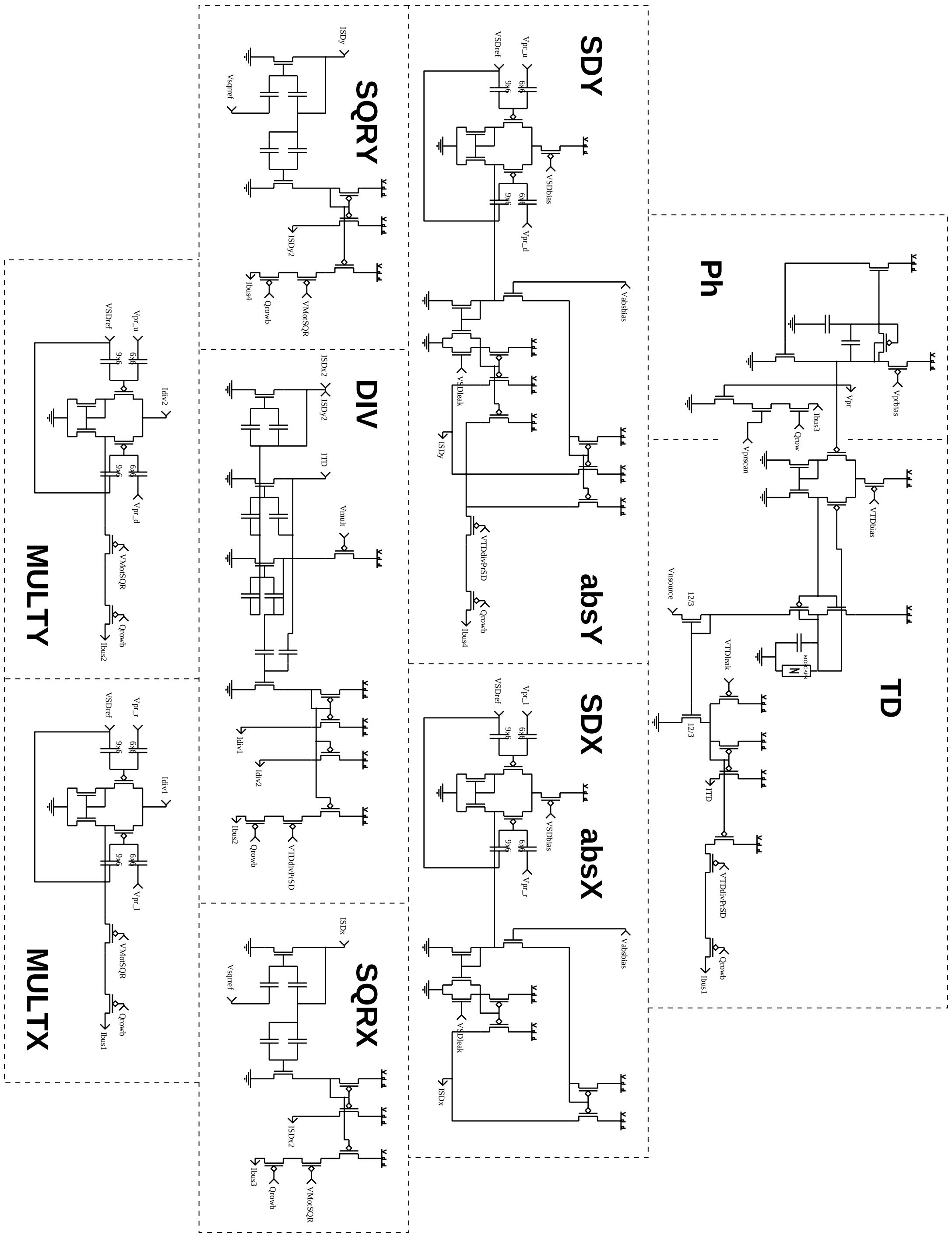


Figure 3.52: Pixel schematic of Velocity2d.

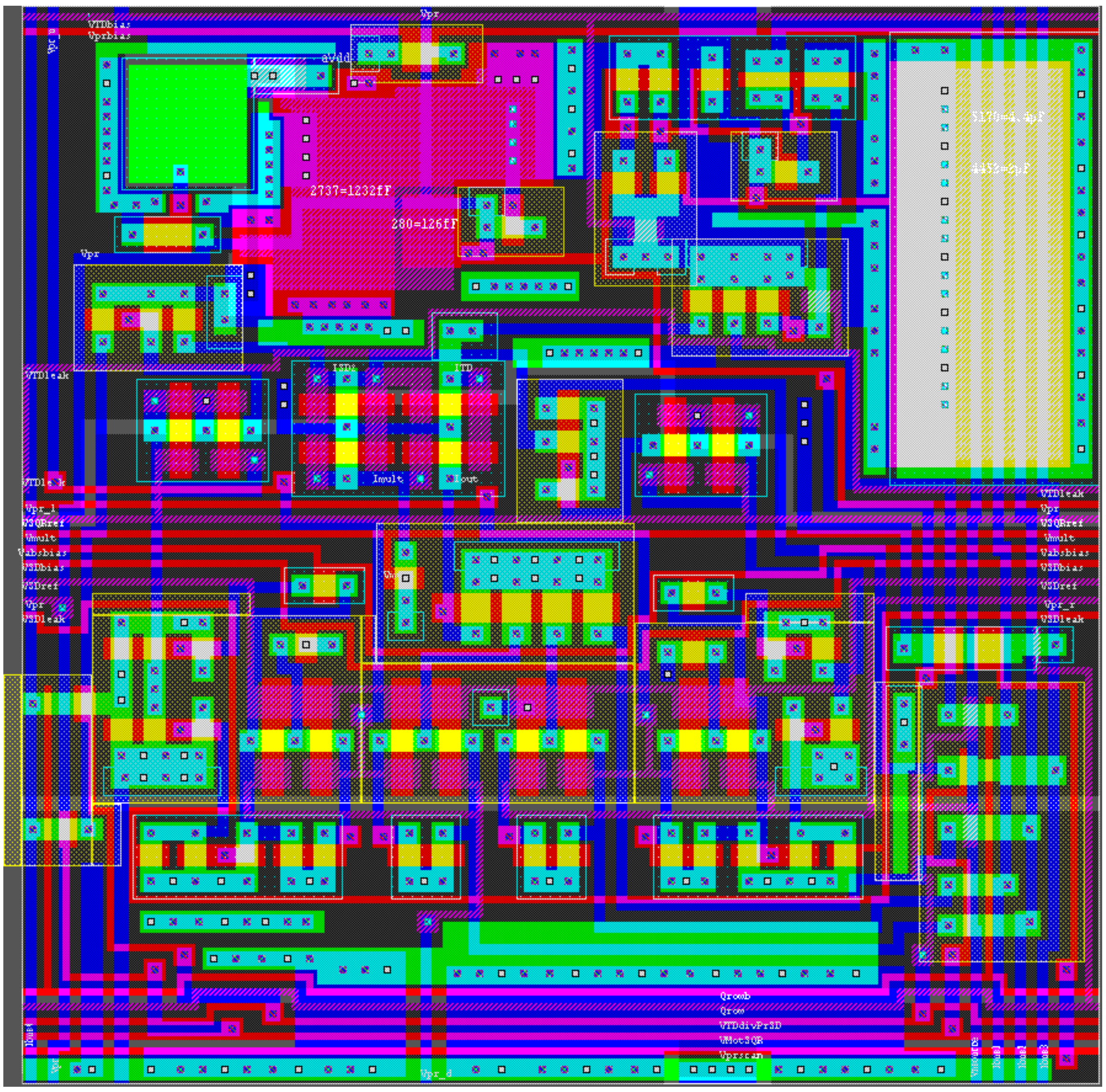


Figure 3.53: Pixel layout of Velocity2d.

3.13 Summary and Discussion

Despite the many challenges associated with an implementation of the gradient method, as discussed in section 3.2, the CMotion1d sensor was found to perform very well. The single pixel output and the summed output of the whole array met the theoretical predictions. The sensor was shown to respond almost linearly to over two and a half orders of magnitude of velocity, down to very slow and up to very fast velocities. This large dynamic range despite a non compressive output representation was achieved because all computation is performed in current mode, where currents from 0 to 100 nA are used and pico amperes are valid signals. Nonlinearities in the temporal and spatial derivatives were modeled by a sensor model using different low pass filter time constants for the photoreceptor and the derivative circuits. This model could explain temporal and spatial derivatives very accurately. The slight saturating nonlinearity in the velocity output could be attributed mainly to above threshold operation and remaining charges on the floating gates. The spatial frequency dependence was as theoretically expected for discrete spatial sampling. The velocity output was shown to be independent of contrast for contrast above 20 %, and going to zero for lower contrasts. The dependence on the stimulus orientation was as theoretically predicted.

Gradient based sensors as compared to feature based sensors have the advantage that no thresholding is necessary. The velocity output of the CMotion1d sensor is generally very robust. For lower and lower contrasts the output degrades continuously. The output might not be as accurate as for high contrasts any more, but the stimulus is not lost entirely all of a sudden. With thresholding, on the other hand, weak stimuli do not generate any signal or, even worse, completely wrong signals are generated.

Problem areas

Several problem areas have been identified with the CMotion1d sensor:

- Even though the CMotion1d sensor operates in current mode, where signals can span many orders of magnitude, the output eventually has to be amplified and converted into voltage domain, where with the given signal to noise ratio one and 1/2 orders of magnitude of velocity can be correctly detected. This velocity range is sufficient for most applications, but a larger range might me desired in some cases. For that purpose two possible solutions exist: First the velocity output instead of being linear could be coded in a compressive way, for example logarithmically increasing with the stimulus velocity, as in the FS sensor [KSK97]. Second the range of velocities to which the CMotion1d sensor is sensitive can almost arbitrarily be set by the amplification bias voltages for the temporal and spatial derivative and the amplification of the output current. If sensitivity to a low velocity range is desired, then the temporal derivative and output current can be amplified through $V_{nsource}$ and V_{mult}. If then much higher velocities are presented, the sensor output will automatically become compressive due to saturation mainly in the temporal derivative and division circuits. Ideally the gain biases were self-adaptive such that the correct operating point would be detected automatically. For example the mean and standard deviation of the detected velocity could be used to determine the DC offset and slope of the function which computes the output from the stimulus velocity.

- The observed gain mismatch in the velocity output between leftward and rightward motion has been traced down to the absolute value circuit and the fact that the photoreceptor has different time constants for rising and falling intensities. The photoreceptor output can be buffered and for the absolute value a better circuit has to be developed.

- As has been shown the spatial frequency dependence of the velocity output, which is undesired, cannot be avoided for any discrete implementation of the gradient method. But it can be ameliorated by increasing the fill factor of the photo active area, by a tighter pixel spacing and by blurring of the lens.

- The nonlinearity of the velocity output with stimulus velocity has been found to be due to the fact that there are different time constants associated with the temporal and spatial derivative circuits. A nonlinear output can also be generated if the dynamic range of the circuits is exceeded (e.g. for too high velocities or too high contrasts for the given bias setting) or if there is charge inequality on the floating gates.

- For the present implementation only one sign of the temporal derivative is used. This was sufficient because both spatial derivative signs and a sign correction circuit were used. Taking in to account also the other temporal derivative sign would buy a longer duty cycle of the output. The difficulties would be the expected mismatch between positive and negative temporal derivative and to keep the pixel size reasonably small.

- For the division by zero problem several cures were found. An effective method to prevent the velocity output from going out of bounds will be even more necessary when both temporal derivative signs are taken into account.

- Even if both temporal derivative signs were taken into account, the duty cycle of the output would still be less than one. The reason is the crossover distortion introduced by the temporal derivative circuit. This means that even for a continuous sine wave stimulus there would be times in the output where periodically the output would report zero velocity.

- A problem inherent to the gradient method is the responsiveness to overall changes in the light intensity in the presence of spatial gradients (flicker). This problem could be treated by spatial low pass filtering and suitable integration over several pixels.

Chapter 4

Gradient2d Sensor: Gradient Based Scheme for 2-D Motion Detection

The computation of the 2-D velocity field on a single chip sensor is a very difficult task, even more so if also high pixel resolutions are desired. Several approaches have been investigated in the past. In some implementations of feature based algorithms the time-of-travel of a feature between two locations on the focal plane is measured. In other sensors the gradient method is implemented using temporal and spatial derivatives. Implementations of both methods either yielded a large pixel size or showed poor performance, or both.

In this chapter a sensor with very small pixels will be described, that computes a 2-D vector field based on temporal and spatial derivatives, and which shows a robust performance. Each individual vector is oriented in the direction of the local gradient of the intensity pattern. The length of each vector is increasing with the local stimulus velocity. Both orientation and length of the vectors are dependent on the spatial frequency and contrast of the stimulus.

The motion vector $\vec{u} = (u_x, u_y)$ is computed by the function:

$$\boxed{\begin{aligned} u_x &= u_o \frac{\partial I}{\partial t} \tanh\left(\frac{\partial I}{\partial x}\lambda\right) \\ u_y &= u_o \frac{\partial I}{\partial t} \tanh\left(\frac{\partial I}{\partial y}\lambda\right) \end{aligned}} \tag{4.1}$$

where $I(x, y, t)$ is the light intensity distribution on the focal plane, $\partial I/\partial x$ and $\partial I/\partial y$ are both spatial derivatives, $\partial I/\partial t$ is the temporal derivative, u_o a scaling constant and λ a constant defining the slope of the linear region of the $\tanh$ function. The spatial derivatives are discretely approximated in the implementation. This equation can be derived by a circuit analysis.

The Gradient2d sensor is based on a 1-D motion sensor developed by Horiuchi [Hor97] [HMKD97], whose advice is gratefully acknowledged.

This chapter is organized as follows: In the next section the Gradient2d sensor and its implementation will be outlined, in section 4.2 the time dependent motion output will be explained, in the following sections experimental results for the motion output dependency on the stimulus velocity, orientation, contrast and spatial frequency will be presented. In section 4.6 the direction selectivity of the Gradient2d sensor for very low contrast stimuli will be presented. The chapter will be concluded with a summary and an outline for future work.

4.1 Circuit and Layout

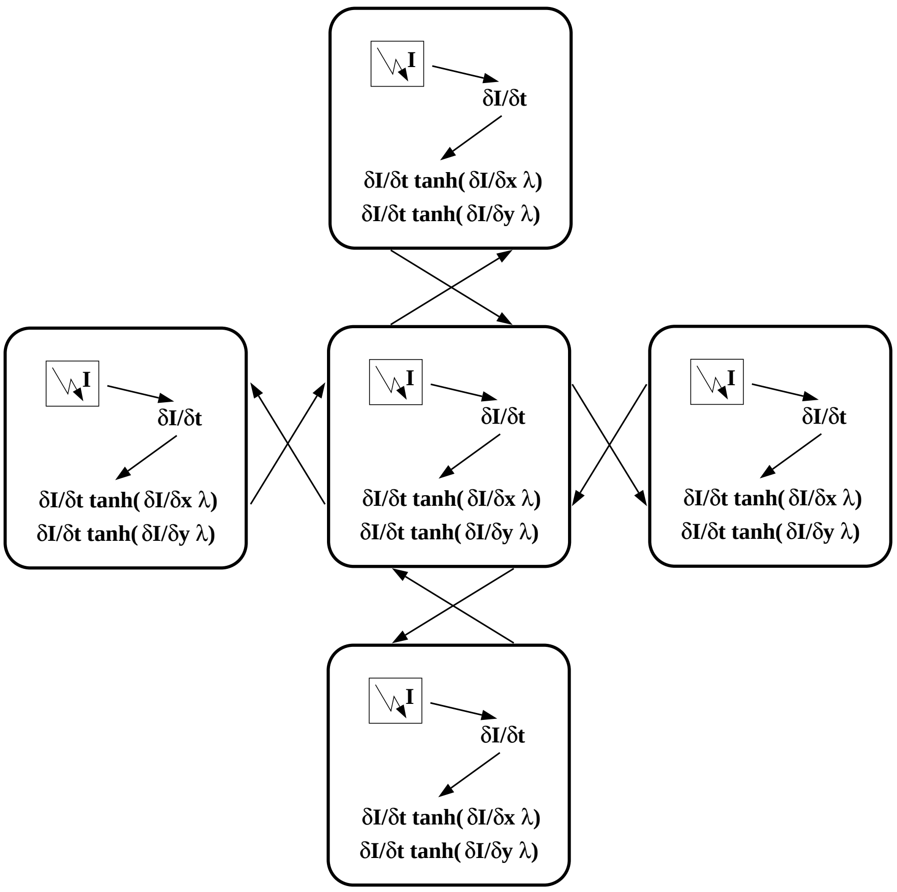


Figure 4.1: Block diagram of the Gradient2d sensor. In every pixel the temporal derivative is computed and multiplied with a tanh function of the spatial derivative. This is done for both spatial dimensions. Every pixel transmits the local light intensity to all four neighbors (indicated by outgoing arrows), and receives the light intensities from there (indicated by incoming arrows). Therefore only two wires are required between every pair of pixels. It can be seen that the Gradient2d sensor consists of two orthogonally oriented 1-D sensors.

The Gradient2d sensor was designed to locally compute a 2-D signal which indicates the direction-of-motion and velocity of a stimulus. In order to achieve a small pixel size, an algorithm was used which required only few computational steps. As can be seen in the schematic of the Gradient2d pixel in Figure 4.2, every pixel only contains a photoreceptor, a temporal derivative circuit and two multiplication circuits, which compute the product of the temporal derivative and a tanh function of the spatial derivative for both spatial dimensions. The spatial derivative circuits are not required for the motion computation and have been added to compute a real time edge map.

The pixels were arranged in a 2-D array such that to all four neighbors the local light intensity is reported and at the same time the light intensities from all four neighbors are received, see the block diagram in Figure 4.1. This communication scheme only requires the use of two wires between every pair of pixels.

Because of the small amount of building blocks required for the motion computation even in 2-D, the pixel could be laid out very compact. In Figure 4.3 the layout of the Gradient2d2 pixel is demonstrated, which has been fabricated in a 2μm process. For this particular version of the sensor the pixel dimensions are $217\mu\text{m}\times 201\mu$m. A later version of the same algorithm has been submitted for fabrication in a 1.2μm process, and a pixel size of only $112\mu\text{m}\times 112\mu$m could be achieved. In the next figure the layout of the entire chip is shown. The 7×7 pixel array can be recognized. At the left and lower borders the scanners are visible, which allow to scan the local motion information from every pixel. The chip layout can be compared with a photograph of the actual chip as shown in Figure 4.22.

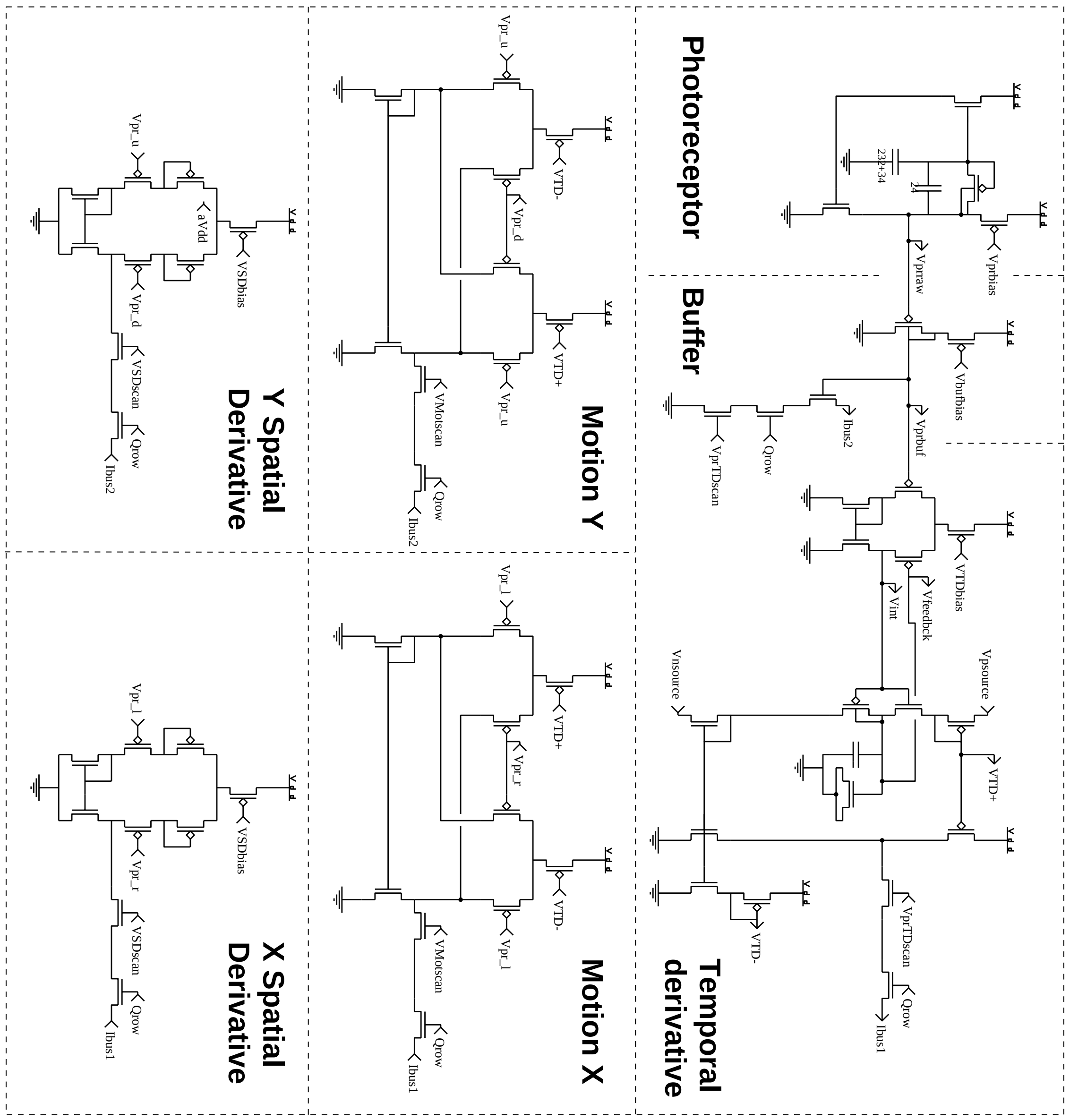


Figure 4.2: Schematic of the Gradient2d sensor pixel. Building blocks are the adaptive photoreceptor, a buffer, the temporal derivative circuit and a circuit which multiplies the temporal derivative with a sigmoidal function of the spatial derivative in both spatial dimensions. The two spatial derivative circuits are not required for the motion computation and are added for independently computing an edge map as an additional feature.

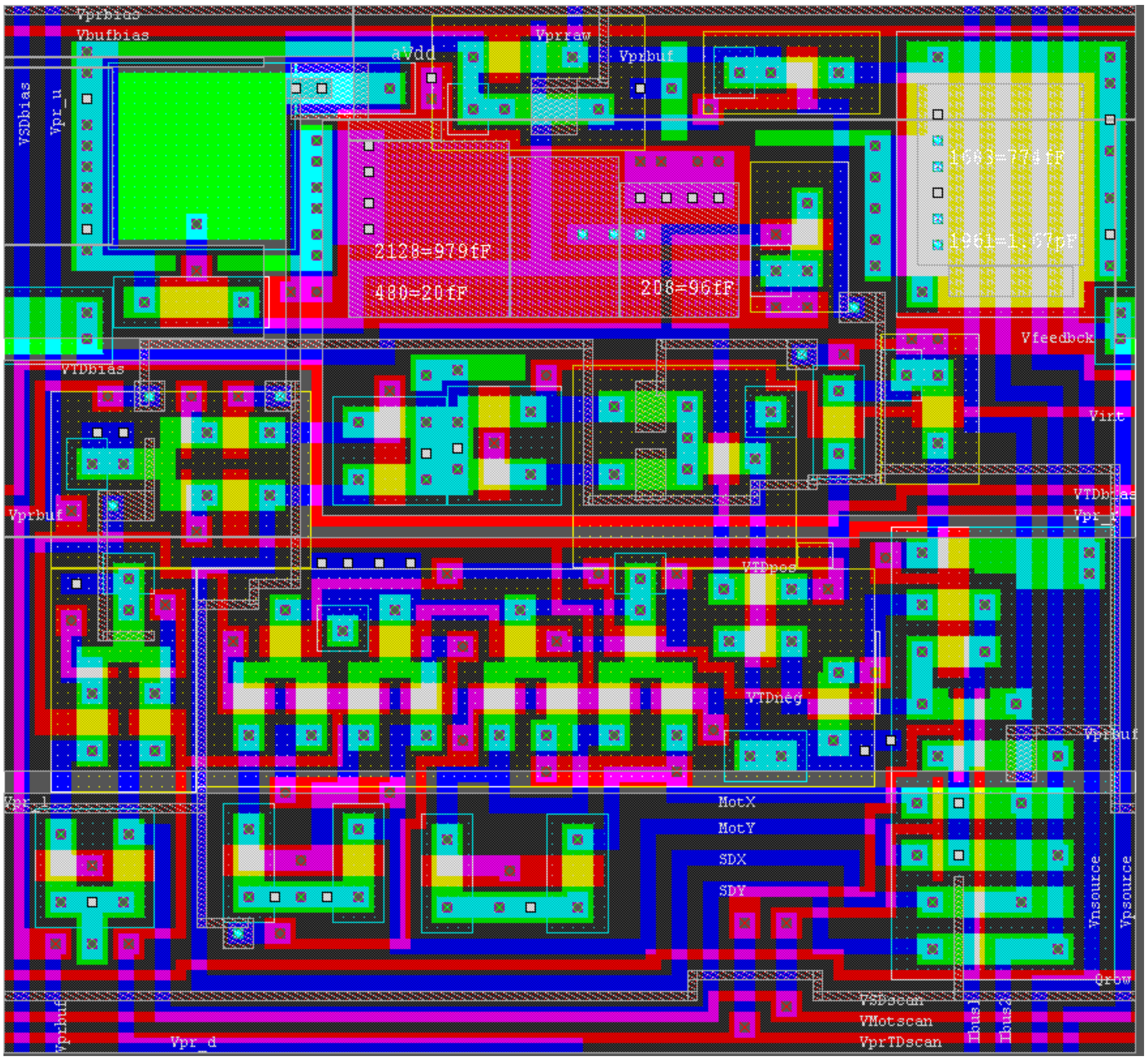


Figure 4.3: Layout of the Gradient2d pixel.

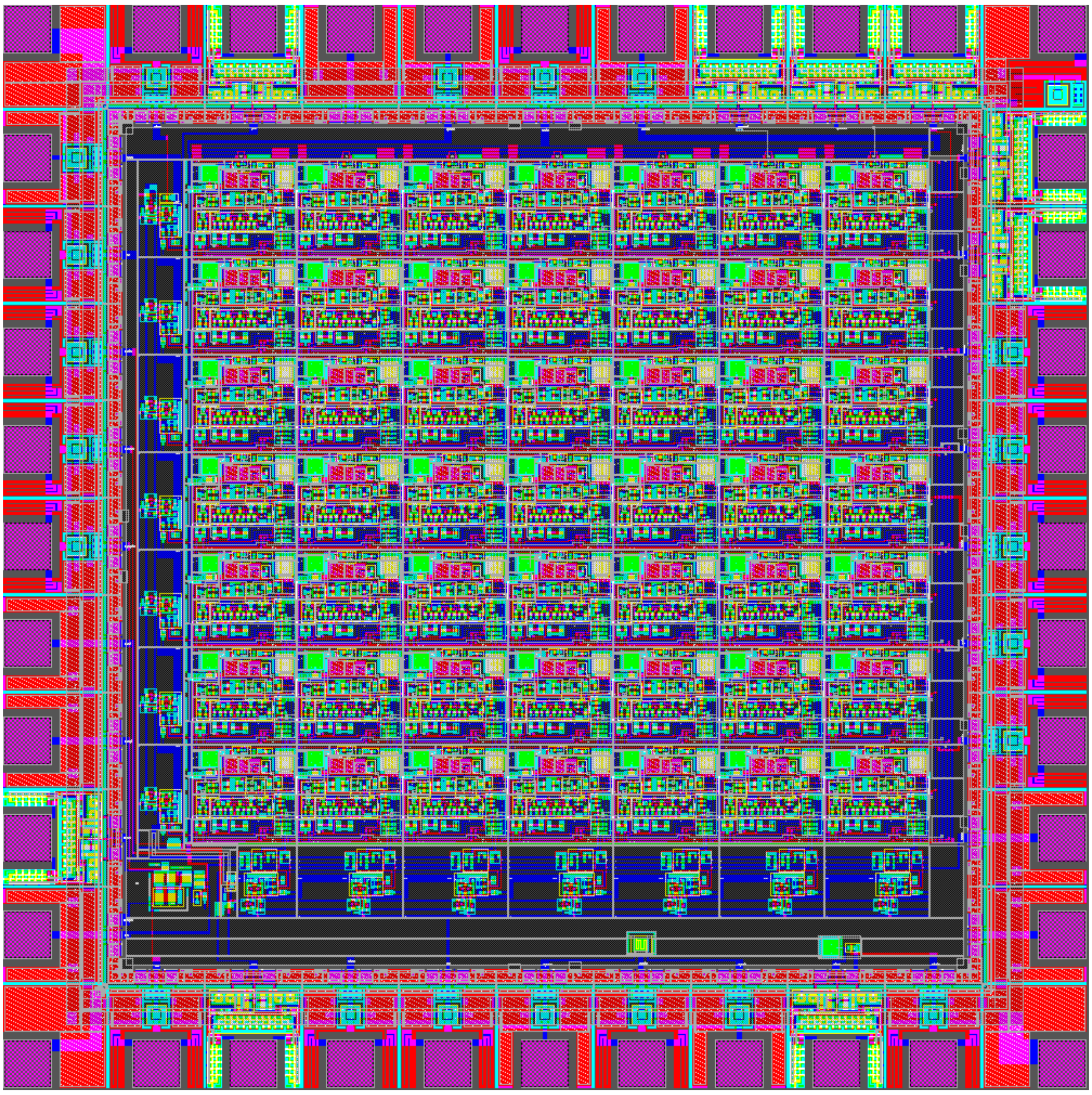


Figure 4.4: Layout of the Gradient2d sensor.

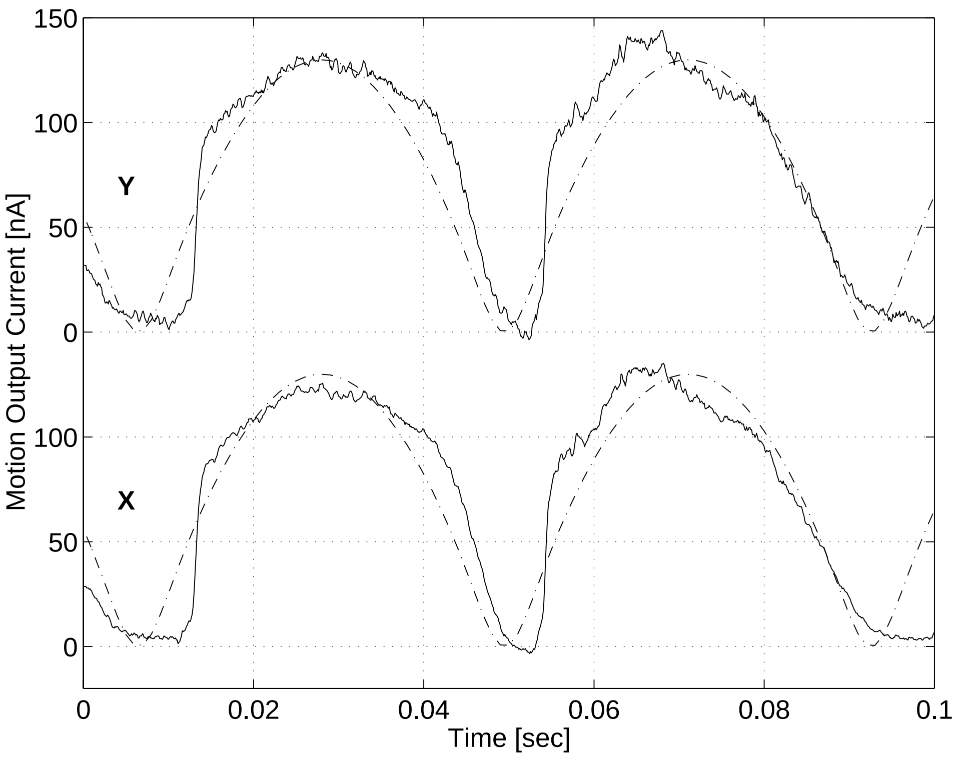


Figure 4.5: Time course of motion signal for sinusoid stimulus. The X and Y output of one pixel was recorded over time for a sine wave stimulus oriented diagonally. One period is shown. It can be seen that the motion output is time dependent, for applications a time average would be taken. The dashed line shows the theoretical expectation.

4.2 Results: Sample outputs

Unlike the gradient method for velocity computation, where a constant velocity output is generated for a stimulus of constant velocity, the Gradient2d algorithm predicts a time dependent motion output for any moving stimulus with non linear gray value pattern. For most of the experiments presented in this chapter sine wave gratings are used as stimulus. From equation 4.1 the expected time dependent motion output for a sine wave grating of amplitude I_o, spatial frequency k_o and moving in the direction φ with velocity v can be derived:

$$\begin{aligned} u_x(\varphi,t) &= v\,u_o\,I_o\,k_o\,\cos(\omega\,t)\,\tanh\left(I_o\,k_o\,\cos(\varphi)\,\lambda\,\frac{\sin(k_o\,\cos(\varphi)\,\Delta)}{k_o\,\cos(\varphi)\,\Delta}\,\cos(\omega\,t)\right) \\ u_y(\varphi,t) &= v\,u_o\,I_o\,k_o\,\cos(\omega\,t)\,\tanh\left(I_o\,k_o\,\sin(\varphi)\,\lambda\,\frac{\sin(k_o\,\sin(\varphi)\,\Delta)}{k_o\,\sin(\varphi)\,\Delta}\,\cos(\omega\,t)\right) \qquad (4.2) \end{aligned}$$

where $\omega = v\,k_o$.

In Figure 4.5 the time dependent motion output for both X and Y channel is shown for one pixel. A sine wave grating of spatial frequency 0.012 cycles/deg, contrast 70 %, velocity 13.55 mm/sec and diagonal orientation was used. It can be seen that X and Y motion output are mutually very

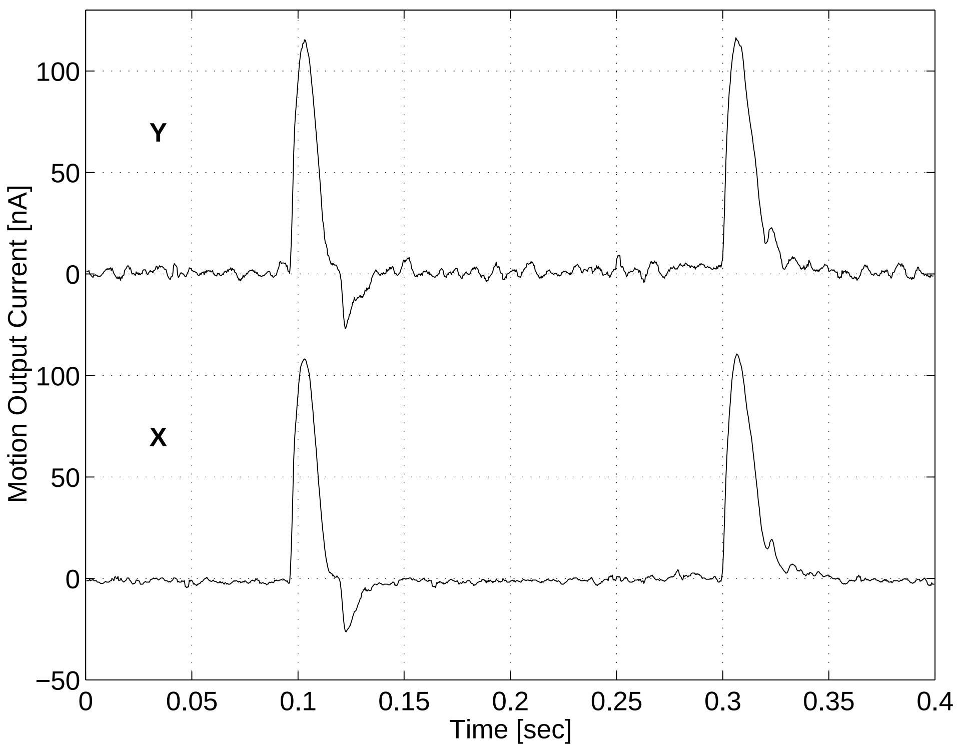


Figure 4.6: Time course of motion signal for square wave stimulus. The X and Y motion output were recorded over time for a sharp black bar on white background moving past the sensor. Both on the bright-to-dark and the dark-to-bright edge large motion output is produced. The short null direction output at the bright-to-dark edge is caused by excess current in the temporal derivative circuit.

similar, the peak heights are similar, and both show a delayed onset, which is due to the crossover distortion in the temporal derivative circuit. The dashed curves are the fit functions according to equation 4.2. Only the qualitative time course can be explained. More test outputs from the pixels are required to explain the fine structure.

The Gradient2d sensor is very suited for graded stimuli because a motion output is generated continuously in time. Only if temporal and spatial changes are visible in the visual scene a motion output is generated. The sensor does not require edges for operation, in fact theoretically for infinitely fast circuits the sensor response to a sharp edge would be a delta peak. Before and after the edge the temporal derivative would be zero and therefore motion output could only be generated at the very edge.

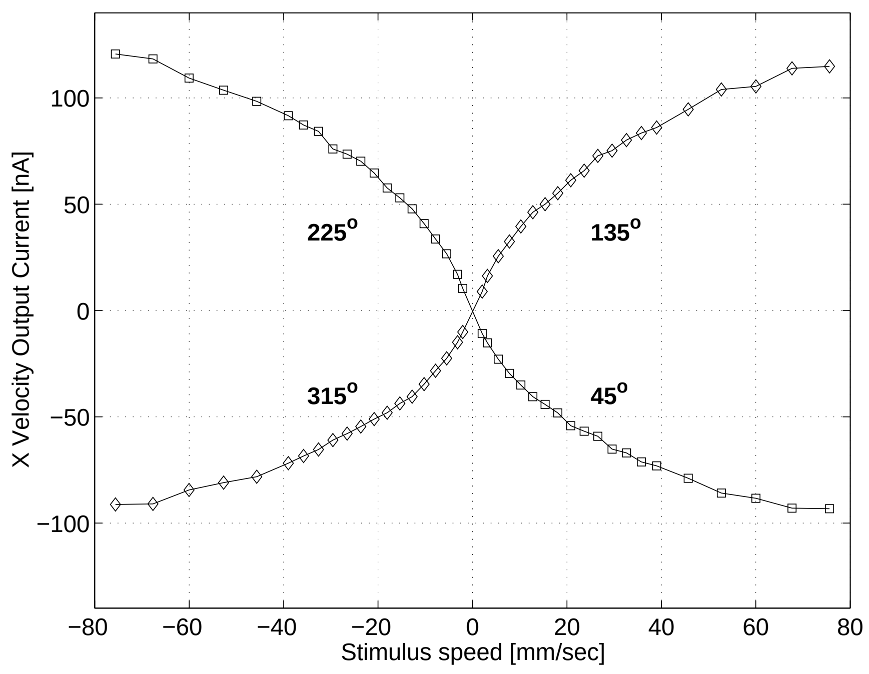
The sensor output for an edge has been recorded. The lens was focussed precisely. Sharp edges are a challenge for gradient sensors, because the sudden strong signal can lead to overshoots in the temporal derivative circuit which can cause null direction output. In Figure 4.6 s sample scope trace is shown for both X and Y motion output. A bright-to-dark and a dark-to-bright edge is visible. Because of the low pass filter behavior an extended output (20 ms) and finite height is obtained. The dark-to-bright edge does not cause any null direction output, whereas an interesting effect can be observed for the bright-to-dark transition. After the edge has been detected, like an echo a small null direction output is produced. If the output were time averaged over more than 50 ms this null direction output would not be perceived. The reason for this glitch lies in the temporal derivative circuit: For a bright-to-dark edge the temporal derivative capacitor can be discharged too much and a small current is required to recharge it, which causes a temporal derivative current of the wrong sign.

4.3 Results: Velocity Dependence

For the purpose of testing the motion output of single pixels it is useful to take the time average of the motion output. In Appendix A.1 it is shown, that for a sine wave input the mean of the motion output is linear in the stimulus velocity.

In the plots of Figure 4.7 the motion output of a Gradient2d sensor pixel is summarized. A sine wave stimulus of 72.5 % contrast and 0.068 cycles/degree spatial frequency was moved over the sensor and the motion vector output of one pixel was recorded and averaged over time for different stimulus speeds. The stimulus orientation was chosen for this experiment in all four diagonals, such that a full comparison between opposite directions of motion and orthogonal directions of motion could be made. Therefore for both the X and Y component four traces were obtained, respectively. The relevant stimulus direction-of-motion is indicated in the plots.

Several observations can be made from the experimental data. First and most importantly the motion output is monotonically increasing with stimulus speed for all directions and for both vector components equally. As expected because the stimulus direction-of-motion was diagonal both motion components have the same absolute value for a given speed. The motion output is not linear with the stimulus speed, but saturates at the high ends. This effect will be discussed below. A gain mismatch between two opposing direction-of-motions can be observed. This effect is caused by device mismatch in the multiplication circuit.



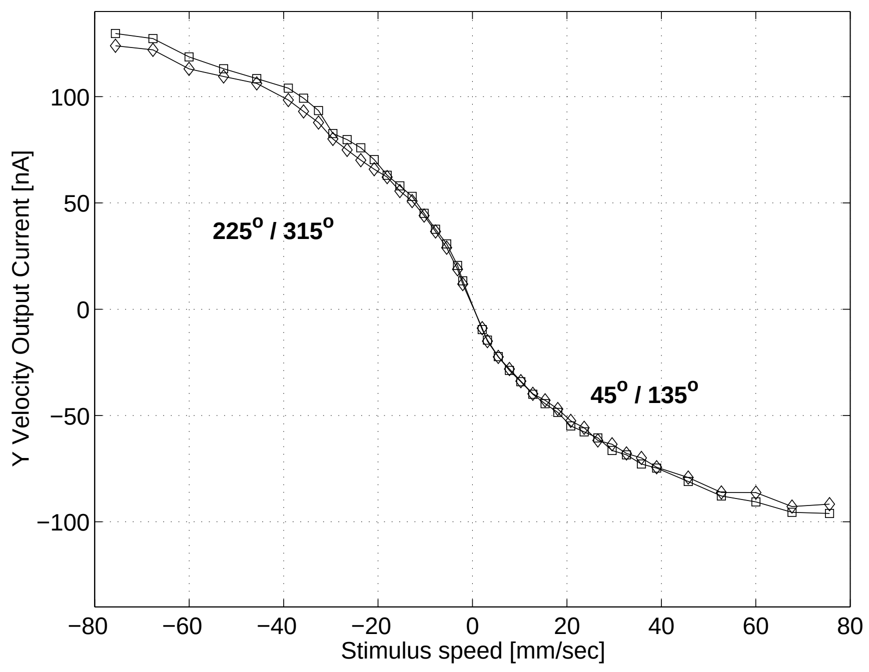


Figure 4.7: Velocity dependence of the Gradient2d sensor motion output for a sine wave with spatial frequency 0.068 cycles/degree and 72.5 % contrast. The orientation of the stimulus was diagonal in 45^o and 225^o (box symbol) and in 135^o and 315^o (diamond symbol). The X and Y component of the motion vector are shown on the top and bottom plot respectively. The motion output is increasing with stimulus velocity, the compressive nonlinearity can be explained with low pass filter behavior in the temporal derivative and multiplication circuits.

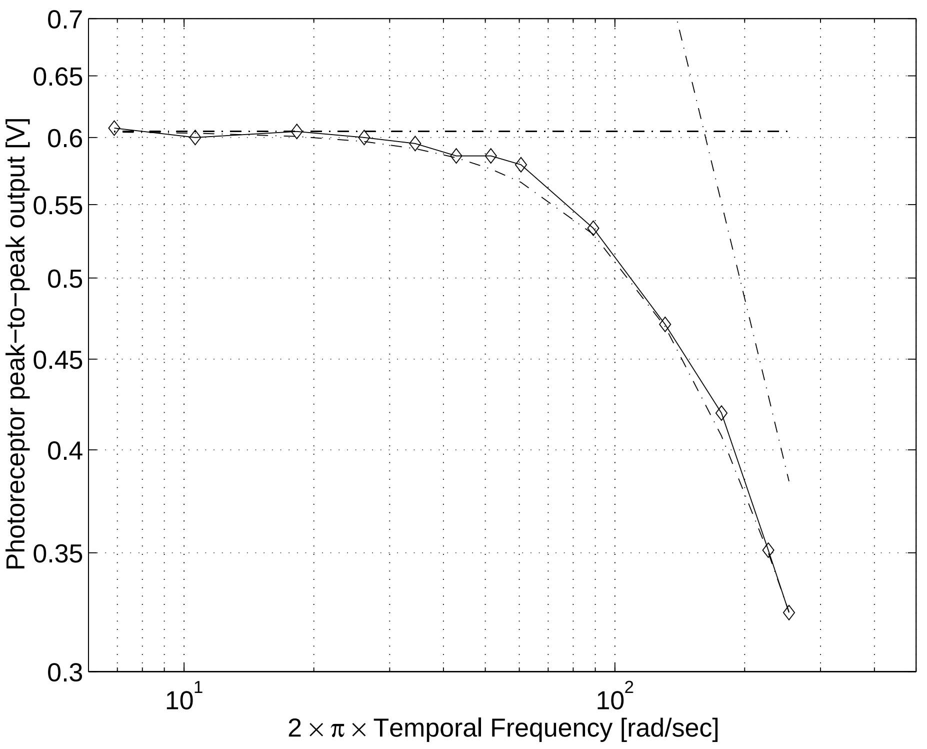


Figure 4.8: Low pass filter fit to photoreceptor output in a bode plot. The peak-to-peak photoreceptor output is plotted over the temporal frequency of the stimulus. For high frequencies the photoreceptor does not follow the input signal and cuts off. From the first order low pass filter fit (dotted curve) a time constant $\tau = 1/161$ Hz is found.

4.3.1 Nonlinearity of the Motion Output

For explaining the compressive nonlinearity of the motion output for high stimulus speeds, temporal low pass filter behavior of circuits have to be taken into account. For a system where a sinusoid input signal is low pass filtered in the photoreceptor with one time constant, where the temporal derivative and multiplication circuit are assumed to be fast (negligible time constant), and where a discrete spatial derivative is taken, the following time dependent function for the motion output $u^{LP}(t)$ for one stimulus direction is found from equation 4.1:

$$u^{LP}(t) = u_o\, I_o\, \omega\, \frac{\cos(\omega\, t) - \omega\, \tau\, \sin(\omega\, t)}{1 + \omega^2\, \tau^2}\, \tanh\left(I_o\, \lambda\, \frac{\sin(k\, \Delta)}{\Delta}\, \frac{\cos(\omega\, t) + \omega\, \tau\, \sin(\omega\, t)}{1 + \omega^2\, \tau^2}\right) \quad (4.3)$$

With keeping $\omega = v\, k$ in mind, it can be seen that for slow stimulus speeds equation 4.2 for an infinitely fast system is gained back. For stimulus speeds v for which $\omega\, \tau > 1$ the sine function terms become dominant, the temporal derivative term saturates and the spatial derivative term approaches zero. In effect the motion output vanished for high stimulus speeds.

In order to quantitatively explain the compressive nonlinearity of the motion output observed in the previous experiments, this equation has been numerically integrated. A time independent signal is obtained which can be compared to time averaged measured motion signal. The time constant of

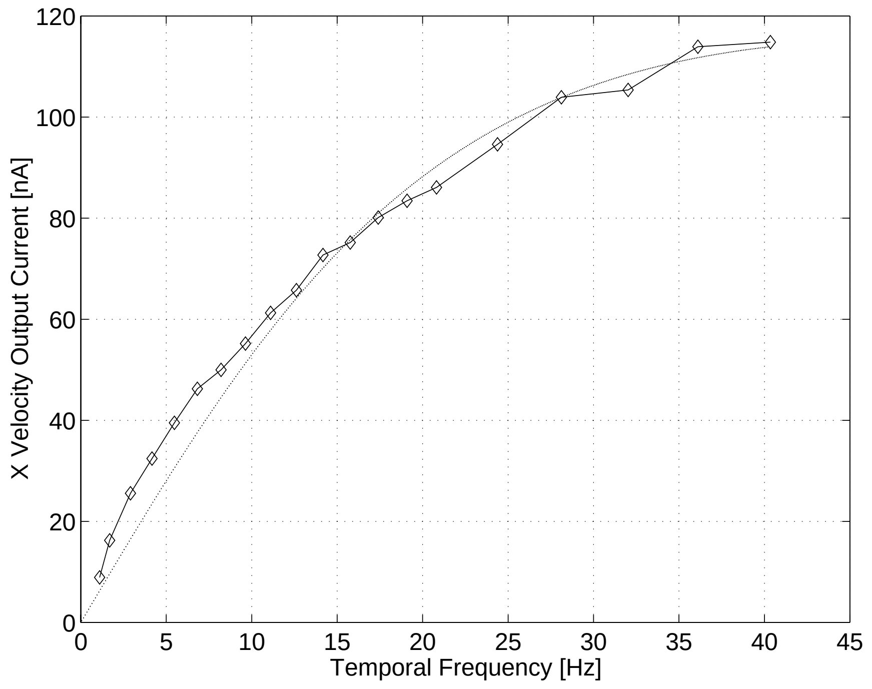


Figure 4.9: Motion output with low pass filter fit. The time average of the measured X motion output for one stimulus direction of motion is plotted. The fit function (dotted line) was obtained from numerically integrating equation 4.3 over one period of the sine wave stimulus. The time constant $\tau = 1/161$ Hz obtained from the photoreceptor signal fit was used. The nonlinearity of the measured motion output can be explained satisfactory.

the photoreceptor τ was found by measuring the peak-to-peak output for sine wave stimulus of fixed spatial frequency moving at different speeds. The first order low pass filter fit (see Figure 4.8) yields a photoreceptor time constant of $\tau_1 = 1/161$ Hz. As can be seen in Figure 4.9 the nonlinearity of the motion output is explained satisfactory by this model.

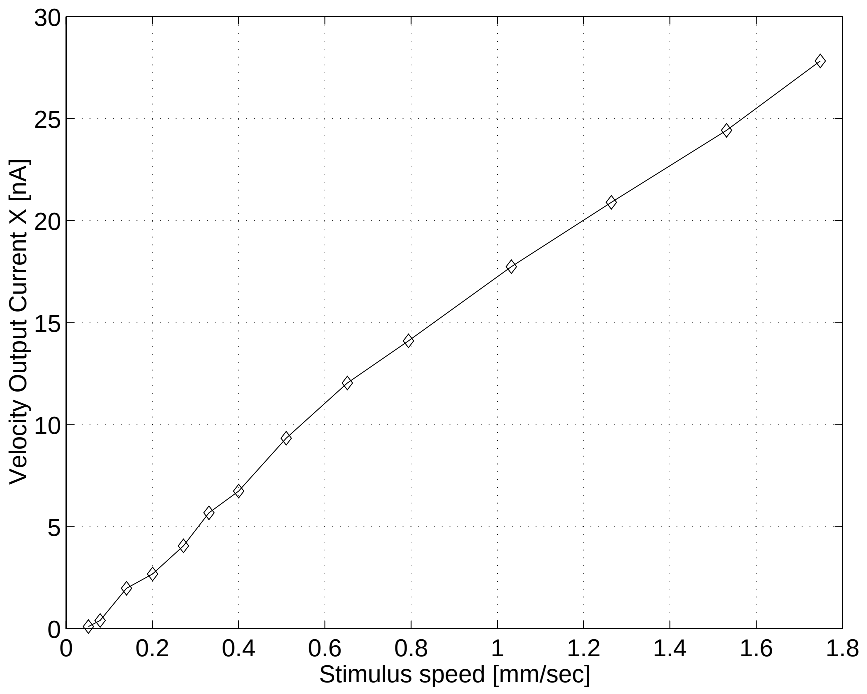


Figure 4.10: Motion output for very slow stimulus speeds. The time averaged motion output is shown for stimulus velocities from 0.052 mm/sec up to 1.75 mm/sec. This experiment is complementary to the velocity experiment presented in the last section, which was done for stimulus velocities above 2 mm/sec. It can be seen that the velocity output is almost linear over the entire range of stimulus speeds. This is because for this low speeds almost no temporal low pass filtering occurs.

4.3.2 Very slow stimulus velocity

The Gradient2d sensor has been tested for very low stimulus velocities, generated with the 'bicycle' device. The parameters and the stimulus were the same as in the last experiment (sine wave of spatial frequency 0.068 cycles/deg, contrast 72.5 %), only the temporal derivative has been amplified by a factor of three in order to be able to read the sensor output with the oscilloscope. The motion output has been averaged over time, as in the last experiment. The result is shown in Figure 4.10. The motion output is almost linear from 0.052 mm/sec to 1.75 mm/sec, which represents far more than one order of magnitude of velocity. The motion output for slow velocities has a high linearity, because the temporal low pass filter behavior does not play a significant role in this case.

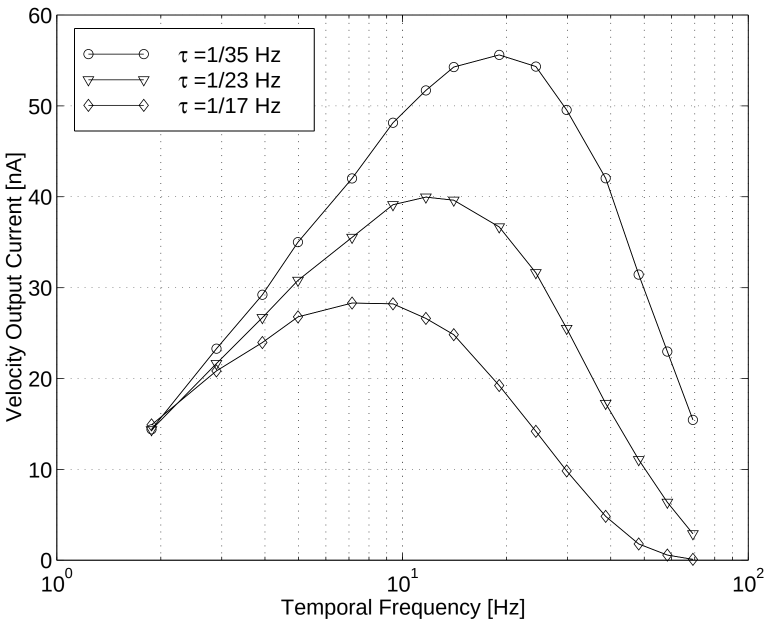
In summary it is found that for both high and slow velocity experiments the Gradient2d sensor computes a motion output which is monotonic in stimulus velocity from 0.052 mm/sec up to 76 mm/sec. This represents a range of more than three orders of magnitude of velocity. Over a wide range the motion output is almost linear with the stimulus velocity.

4.3.3 Velocity tuning curve

Reichardt and colleagues developed a correlation-based model of motion detection that explains the response of the fly visual system to moving stimuli. The Reichardt motion detector works by correlating (by means of multiplication) the response of one photoreceptor to the delayed response of an adjacent photoreceptor, see [Rei87]. Qualitatively the peak output of such an elementary motion detector (EMD) is largest for a velocity which is matched to the delay, and falls off to both higher and smaller velocities. Analog VLSI implementations of the Reichardt model have been presented [HK97a].

In this section it is shown that the Gradient2d sensor can produce qualitatively similar velocity tuning curves to those obtained with the Reichardt model. For that purpose the low pass filter behavior of the photoreceptor is exploited. As was derived in equation 4.3, the temporal derivative of a low pass filtered signal saturates, but the spatial derivative of a low pass filtered signal vanishes for high stimulus velocities. Therefore the time average of the product of those two terms first increases with velocity, reaches a maximum and then decreases for high velocities.

In Figure 4.11 experimental results are shown. A sine wave grating of fixed spatial frequency 0.12 cycles/deg, 70 % contrast and direction-of-motion in X^+ direction was moved at different speeds. The time average of the motion output is plotted for three different time constant settings on the photoreceptor. For each of the three curves the expected velocity tuning can be observed. Additionally for the different low pass filter time constants the peak motion output shifts. Longer time constants cause a shift to smaller velocities, and produce an overall attenuated output.



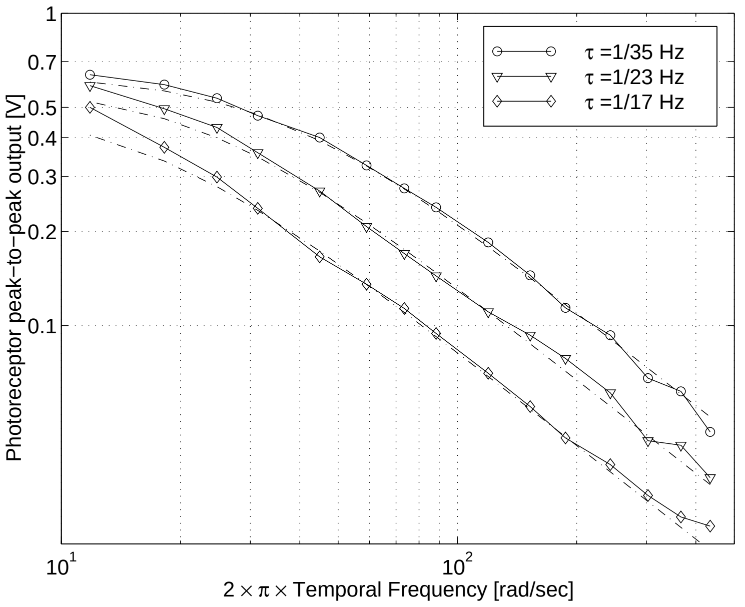


Figure 4.11: Velocity tuning curves. The top figure shows the measured time averaged motion output of the Gradient2d sensor for different stimulus velocities. The smallest curve (diamonds, $\tau = 1/17$ Hz) has the longest time constant and is shifted to the left, the largest curve has its peak shifted to the right (circles, $\tau = 1/35$ Hz), the middle curve has a time constant $\tau = 1/23$ Hz. These time constants were determined with first order low pass filter fits (dashed lines) to the the photoreceptor input, see bottom figure. With just one low pass filter and one multiplication the Gradient2d sensor can produce an output qualitatively similar to a Reichardt detector.

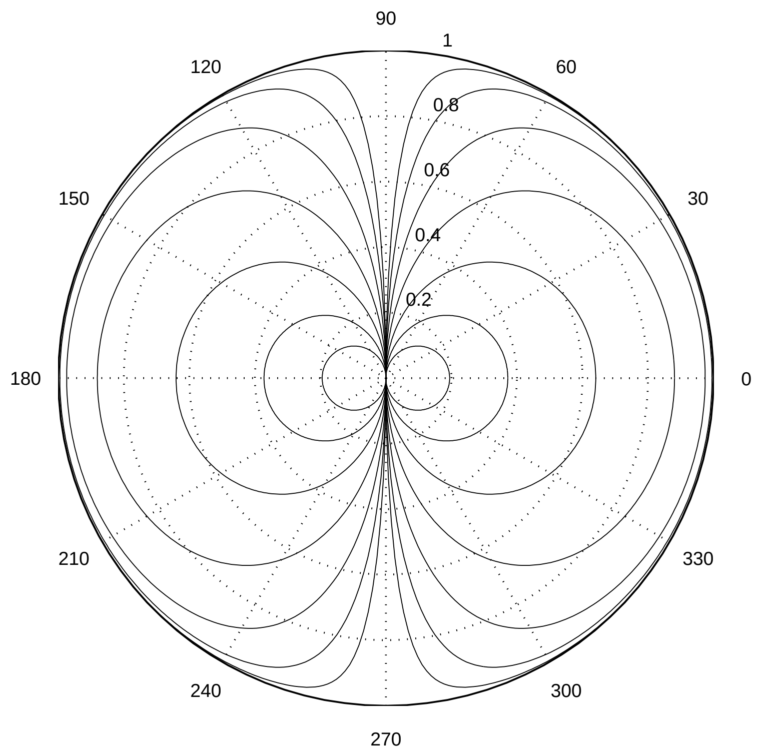


Figure 4.12: Theoretical orientation tuning curves. The X output of the motion vector is plotted for all stimulus orientations according to equation 4.2. The scaling constant $I_o^\Delta\,\lambda$ in the argument of the tanh function is varied in factors of 2. For small scaling constants a cosine tuning curve with small amplitude is obtained, whereas for large scaling constants the tanh saturates and an almost digital direction of motion output is generated.

4.4 Results: Orientation tuning curve

From equation 4.2 in can be seen that for a sine wave stimulus both motion vector components depend mutually different on the time for all angles $\varphi \neq n\,\pi/4; n \in \mathbb{N}$. Therefore the motion vector angle can vary periodically over time. The angular variation, though, is small and vanishes for $I_o\,k_o\,\lambda \ll 1$, i.e. for low contrasts and low spatial frequency input.

Generally for an orientation tuning curve the magnitude of the X and Y component of the motion vector is measured for different stimulus direction-of-motions, whereas the stimulus speed, contrast and spatial frequency is kept constant. Theoretical tuning curves were obtained based on equation 4.2. Sine wave stimuli of different gradients were assumed for the tuning curve displayed in Figure 4.12. For the time average a numerical integration was performed.

In Figure 4.16 experimental data from the time averaged motion output is shown. A 70 % contrast sine wave was moved over the sensor, while spatial frequency and speed were kept constant. As expected a distortion from a sine or cosine tuning curve can be observed. The dashed line shows the fit, which was obtained from the numerically integrated time average of equation 4.2.

Two limiting cases can be considered. First for weak enough contrasts and for circuit implementations with small slope λ (wide linear range of the tanh function) the motion output is approxi-

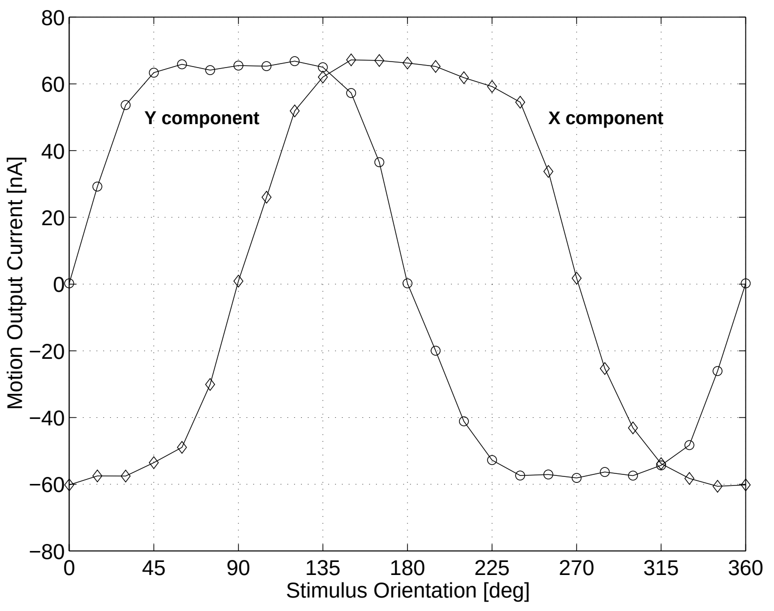


Figure 4.13: Measured orientation tuning curve for high contrast sine wave stimulus (contrast 70%). The time averaged motion output is plotted for stimulus angles from 0^o to 360^o. The tuning curve is distorted from a cosine and sine shape because in the multiplication circuit the spatial derivative is out of the linear range for high contrast stimuli. As can be seen there is a 10% gain mismatch in the motion output between opposing directions of motion.

mately computed by a multiplication of temporal and spatial derivatives. For a stimulus of constant speed and spatial frequency the orientation tuning curve would therefore be

$$\begin{aligned} \bar{u}_x^t(\varphi) &= v\,\frac{u_o}{2\,\lambda}\,(I_o\,k_o\lambda)^2\,\cos(\varphi) \\ \bar{u}_y^t(\varphi) &= v\,\frac{u_o}{2\,\lambda}\,(I_o\,k_o\lambda)^2\,\sin(\varphi) \end{aligned} \tag{4.4}$$

where the factor 2 stems from the time average over the $\cos^2(\omega\,t)$ function. This case was experimentally observed with a low contrast sine wave stimulus and is shown in Figure 4.17. The sine wave had a contrast of 22%.

If on the other hand the factor $I_o\,k_o\,\lambda > 1$, e.g. for high contrast stimuli, then the tanh function saturates and an approximately binary output is computed for both motion components:

$$\begin{aligned} \bar{u}_x^t(\varphi) &= v\,\frac{2}{\pi}\,\frac{u_o}{\lambda}\,(I_o\,k_o\lambda)\;\sigma(\cos\varphi) \\ \bar{u}_y^t(\varphi) &= v\,\frac{2}{\pi}\,\frac{u_o}{\lambda}\,(I_o\,k_o\lambda)\;\sigma(\sin\varphi) \end{aligned} \tag{4.5}$$

where the signum function $\sigma(.)$ is defined as -1 for arguments<0 and +1 for arguments>0, and where the prefactor $2/\pi$ derives from the time average of the absolute value of the cosine function. Only the prefactor for the motion output changes for high contrast stimuli other than sine waves. This binary output was experimentally shown with a high contrast stimulus consisting of black bars

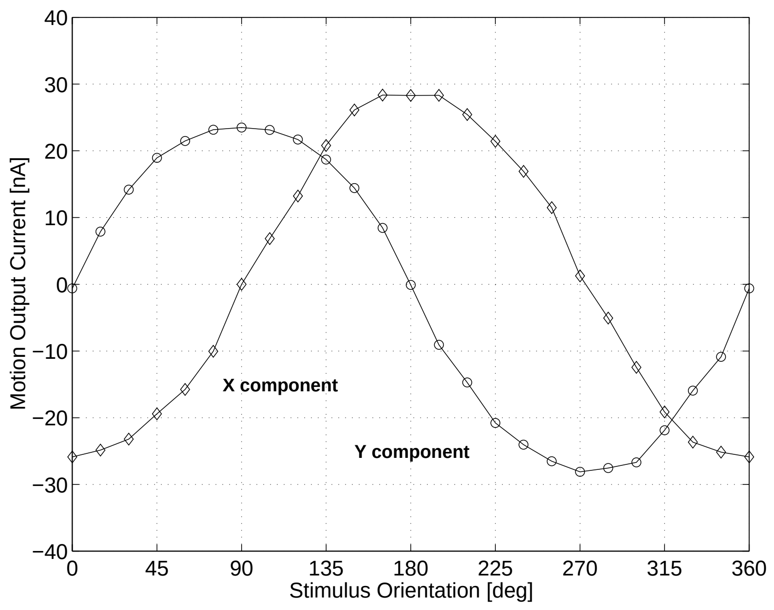


Figure 4.14: Measured orientation tuning curve for low contrast sine wave stimulus (contrast 22%).

on a white background. The lens was focused accurately. In Figure 4.15 it can be seen that for stimulus directions slightly off the rectangular directions the motion output digitally indicates the direction-of-motion. This behavior reminds to the orientation tuning curve obtained with the FTC sensor, see Figure 5.15.

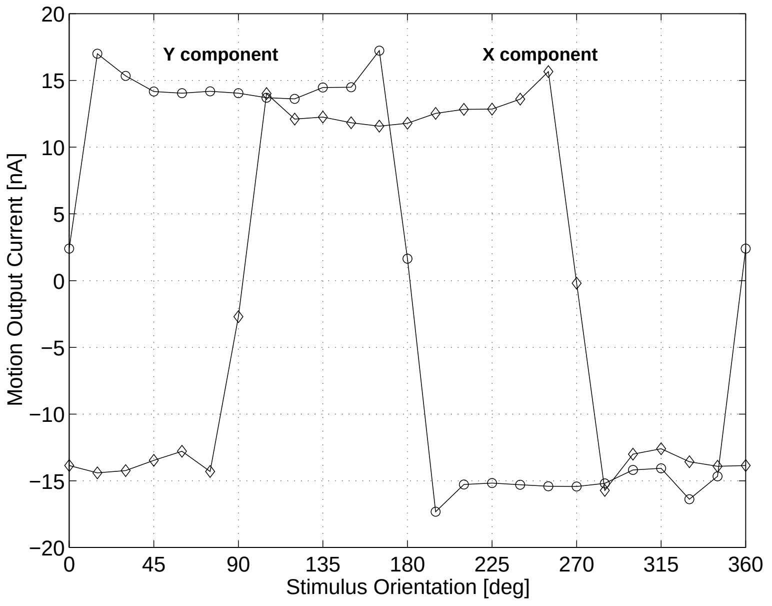
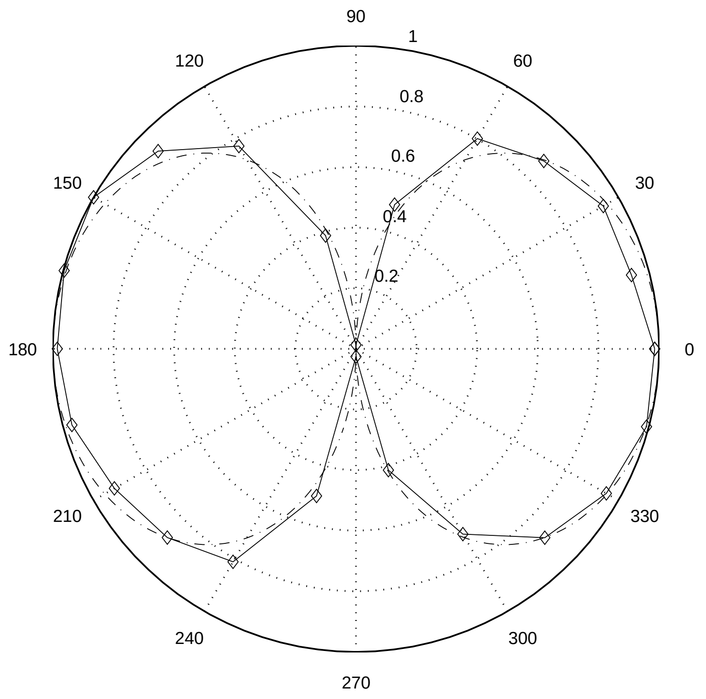


Figure 4.15: Measured orientation tuning curve for sharp edge stimulus. The time averaged motion output is plotted for stimulus angles from 0^o to 360^o. The tuning curve is almost digital, because for stimulus orientations just slightly off the grid orientations the multiplication result is fully saturated.



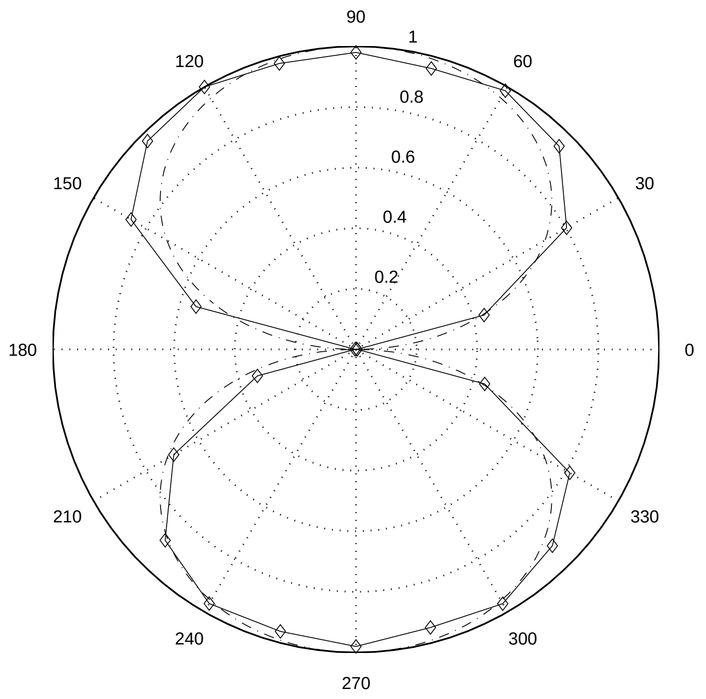
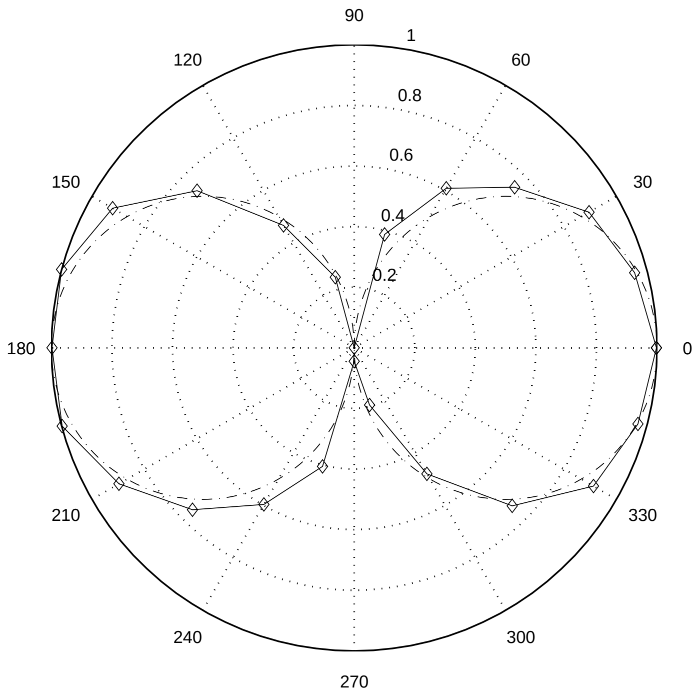


Figure 4.16: Measured orientation tuning curve with fit for high contrast sine wave stimulus (contrast 70 %). X and Y motion output are shown in the top and bottom plot together with the fitted curve (dashed line). If the nonlinearity of the tanh function is taken into account, the measured motion output can be explained very well. The scaling constant $I_o^\Delta \lambda$ in equation 4.2 was found to be 1.6. A gain mismatch between opposing directions of motion of 10% has been compensated for.



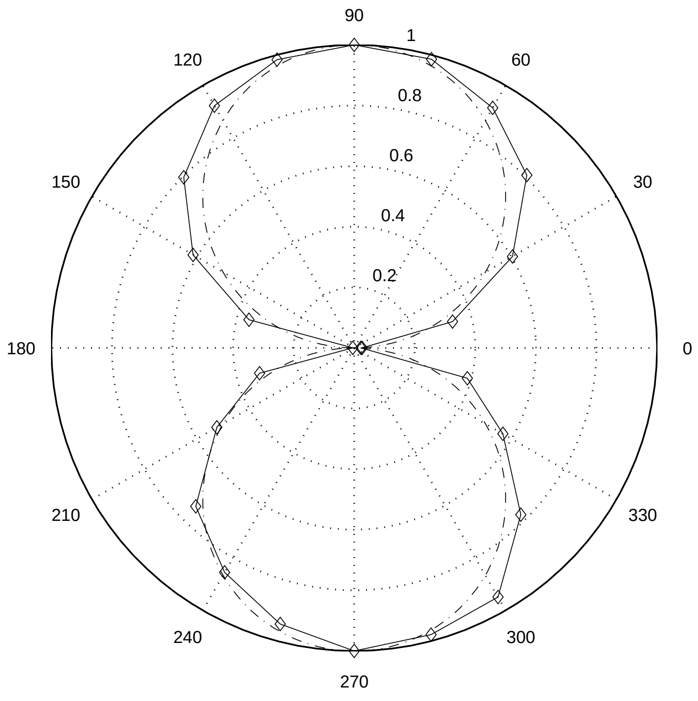


Figure 4.17: Measured orientation tuning curve with fit for sine wave stimulus of 22% contrast. The X and Y motion outputs are plotted on the top and bottom plot, respectively. As can be seen, for low contrast stimuli the tanh function can be linearly approximated and a pure cosine and sine tuning curve is obtained. The dashed line shows the theoretical expectation for low contrasts according to equation 4.4. A gain mismatch between opposing directions of motion of 10% has been compensated for.

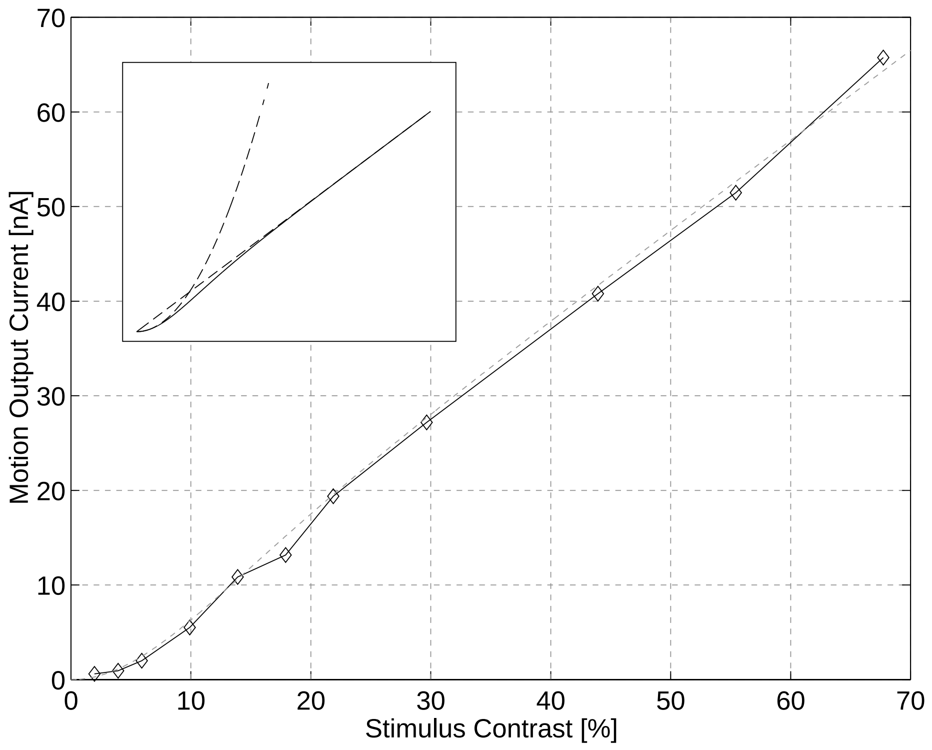


Figure 4.18: Measured contrast dependence of the motion output. As theoretically expected, for low contrasts the time averaged motion output is quadratically dependent on the stimulus contrast, whereas for high contrasts the dependency is linear. In the inset the quadratic and linear fit functions are shown, the dashed line in the main plot indicates the true fit including the transition between quadratic and linear dependency.

4.5 Results: Contrast Dependency

As has been shown in the last section, for low contrasts the motion output is expected to be quadratically dependent on the contrast (see equation 4.4), whereas for high contrasts the motion output should be linearly dependent on the contrast:

$$\begin{aligned} \bar{u}^t &\propto I_o^2 \text{ ; low contrast} \\ \bar{u}^t &\propto I_o \text{ ; high contrast} \end{aligned} \tag{4.6}$$

This theoretical result was confirmed by experiment. Sine wave stimuli of equal spatial frequency, speed and orientation but different contrasts were used, and the motion output was recorded and averaged over time. The result for one direction-of-motion for the X component is shown in Figure 4.18. Both the quadratic and the linear region can be observed.

4.6 Results: Direction Selectivity

In the last section the sensor response has been investigated for different stimulus contrasts. In this section a closer look will be taken on very low contrasts stimuli. Naturally for very low contrasts the

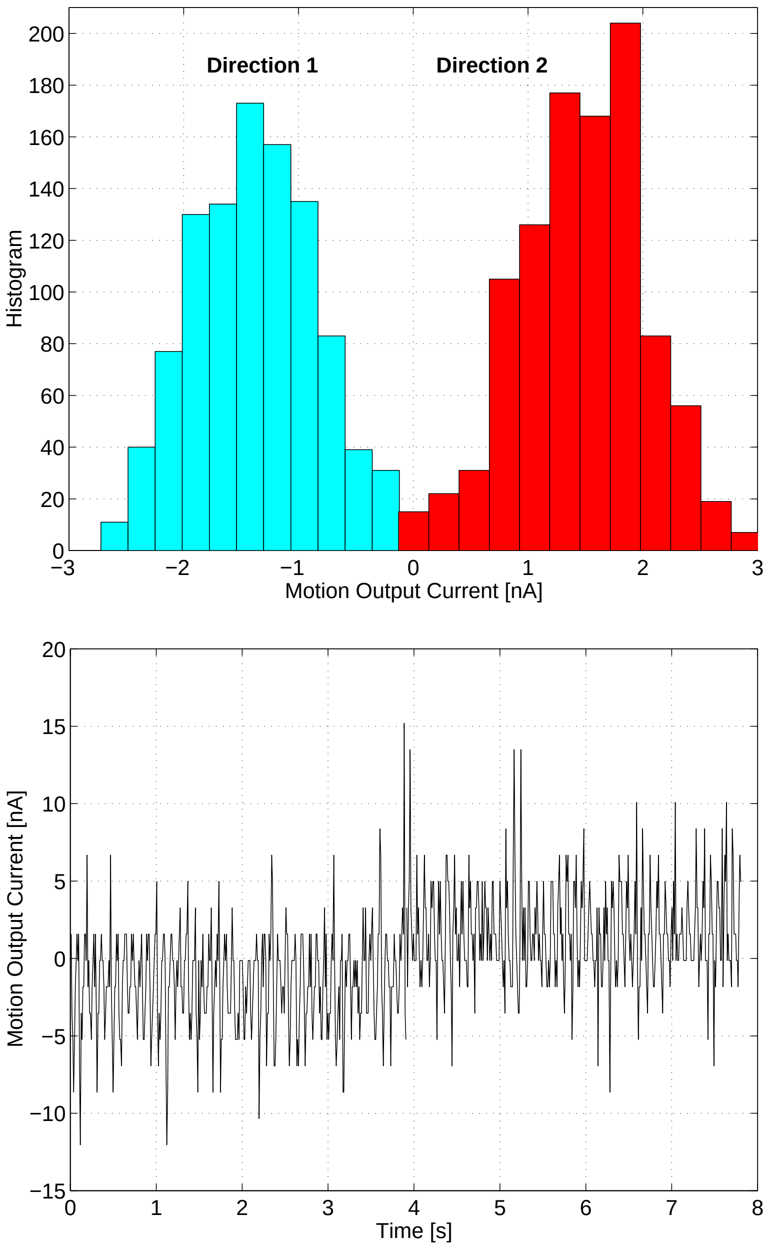


Figure 4.19: Direction selectivity for 4 % contrast stimulus. On the top figure a histogram of the sensor motion output for X direction is histogrammed for a 4 % contrast square wave grating moving in X^+ and X^- direction. The mean values are significantly different at ± 1.42 nA. The motion output was averaged over 100 ms. On the bottom figure the actual motion output is shown, where for t <4 sec the stimulus moves in one direction and for t >4 sec in the other direction. The change in direction can be seen.

signal approaches the noise level, and therefore the reliability of one single reading from the sensor is low. A very important question is, if a short time average of the sensor motion output indicates the correct direction-of-motion.

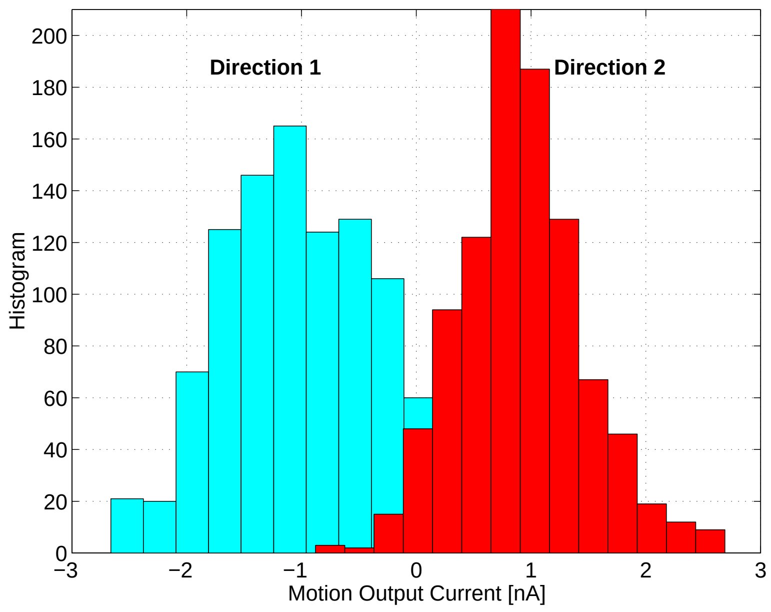


Figure 4.20: Direction selectivity for 2 % contrast stimulus. For this experiment an even lower contrast of 2 % was used. Although the signals become weaker (± 0.95 nA), a discrimination is still possible.

In order to find out for how low contrasts the Gradient2d sensor can discriminate the stimulus direction-of-motion, the sensor motion output for a 4 % and a 2 % stimulus was recorded over time and averaged over 100 ms. The velocity was 10.7 mm/sec for both experiments. The bias settings were not changed as compared to the high contrast experiments. The lowest contrast that could be generated was 2 %. Because this was done using the lightest gray value, no sine waves could be generated, and the stimulus was a square wave grating. A 2 % contrast is barely visible with the eye. For the 4 % contrast stimulus in Figure 4.19 the histogram of the motion output values is shown together with the actual motion output which was low pass filtered over 100 ms. In the histogram as well as in the motion output the two direction-of-motions can clearly been distinguished. There is almost no overlap of both directions in the histogram. In the following Figure 4.20 a 2 % contrast stimulus was used. Although both directions come closer, a distinction is clearly possible.

This experiment is very encouraging, because is shows, that even for very low contrasts the Gradient2d sensor computes the correct direction-of-motion.

4.7 Results: Spatial Frequency Dependence

The spatial frequency dependence of the motion output of the Gradient2d sensor has been measured for sine wave gratings of equal contrast, speed and orientation, with spatial frequencies varying from 0.01 cycles/degree up to 1 cycle/degree. The result is plotted in Figure 4.21. As in the previous sections the time average of the motion output is shown.

From equation 4.1 for the motion vector output the expected spatial frequency dependence can

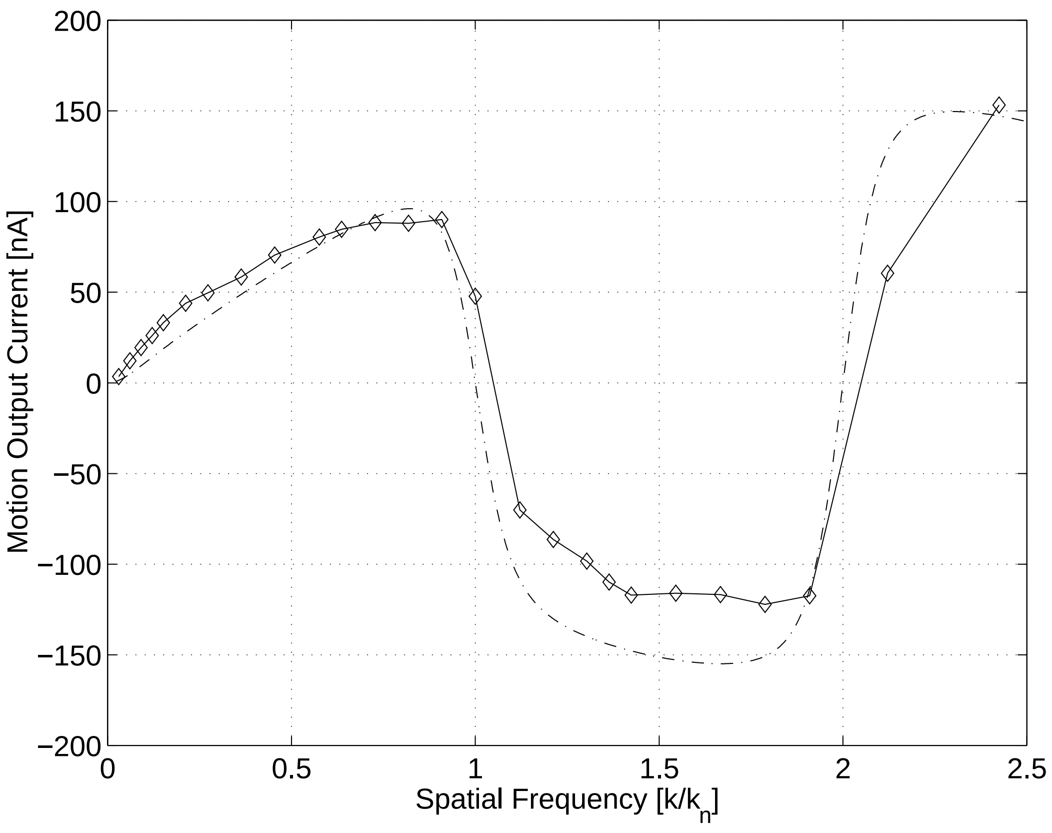


Figure 4.21: Measured spatial frequency dependence of the motion output. The time averaged motion output for sine wave stimuli of same velocity (5.5 mm/sec), same contrast (70 %) and same orientation is plotted for spatial frequencies from 0.01 cycles/degree to 1 cycle/degree. The theoretical expectation considering the discrete spatial derivative and low pass filtering in the photoreceptor is plotted as dashed line. Both curves are qualitatively similar and aliasing can be observed at the predicted spatial frequencies.

be derived. The fact that a discrete spatial derivative is taken has to be considered as well as the low pass filter behavior of the circuits. This is especially important for high spatial frequencies. The photoreceptor and the multiplication circuit was lumped together for the purpose of this experiment. The Nyquist frequency, for which spatial aliasing is expected to occur, was determined from the layout of the sensor: $k_n = 2\pi/2\Delta$ with $\Delta = 217\,\mu$m the inter pixel spacing in the direction of the stimulus. The time dependent motion output was then numerically integrated, the resulting theoretical curve for the motion output was plotted together with the experimental data as dashed line. The fit function has besides a scaling factor two parameters: One parameter for the width of the tanh function and one low pass filter time constant.

As can be seen the measured data shows a similar behavior as the theoretical expectation. The motion output increases with increasing spatial frequency until shortly before the aliasing point, where the output crosses zero and reverses sign. The second aliasing point can even be observed, where the motion output becomes positive again. A very favorable feature of the Gradient2d algorithm as compared to implementations of the true gradient method, where the temporal derivative is divided by the discrete spatial derivative, can be observed: The motion output increases almost linearly until shortly before the aliasing point, and then approaches zero, unlike in the gradient method, where the velocity output goes to infinity at the aliasing point.

There are differences between experimental data and theoretical expectation. For very low spatial frequencies theory predicts a quadratic increase of the motion output, which is not observed. The subsequent increase of the motion output should be linear almost until the aliasing point. The experimental data shows a slightly compressive nonlinearity. In order to explain this phenomenon, in the next version of the Gradient2d sensor more test outputs will be implemented. In fact, the compressive nonlinearity is a desirable feature, because this way the motion output exceeds the noise level earlier for very low spatial frequencies and is less dependent on the spatial frequency subsequently.

4.8 Summary and Future Work

The Gradient2d sensor responds to a wide range of stimulus velocities and contrasts, shows excellent sensitivity to low contrasts, the direction-of-motion output is monotonic with stimulus velocity and continuous in the angular space, the overall behavior is very robust and the sensor is easy to handle with no critical biases. These features suggest the use of the Gradient2d sensor for a front end device in high level vision systems. For the specific motion output of the Gradient2d sensor new algorithms should be developed, because as in many biological sensory systems, the sensory output is dependent on more than just the preferred feature. For example the motion output of the Gradient2d sensor is not only dependent on the stimulus velocity, but also on the stimulus spatial frequency and contrast. The motion vector orientation is parallel the the spatial gradient for lower contrast stimuli, for high stimulus contrasts the motion vectors have a bias towards diagonal orientations. Traditionally high level machine vision algorithms based on motion analysis have been developed under the assumption, that the true velocity field was given. It is suggested here that high level vision tasks can be performed on inputs of higher dimensionality, such as the input from the Gradient2d sensor.

Segmentation from motion can be taken as an example. Traditionally discontinuities in the velocity field were taken for an object boundary. But the motion vector field obtained from the Gradient2d sensor would show discontinuities at the same locations, with the exception of rare occasions when the spatial frequency and contrast dependence of the motion output would cancel out the discontinuity.

Larger arrays of the Gradient2d sensor are in fabrication. Pixel resolutions of 15×15 were achieved on a ‘tiny’ chip, with which high level vision algorithms can already be tested.

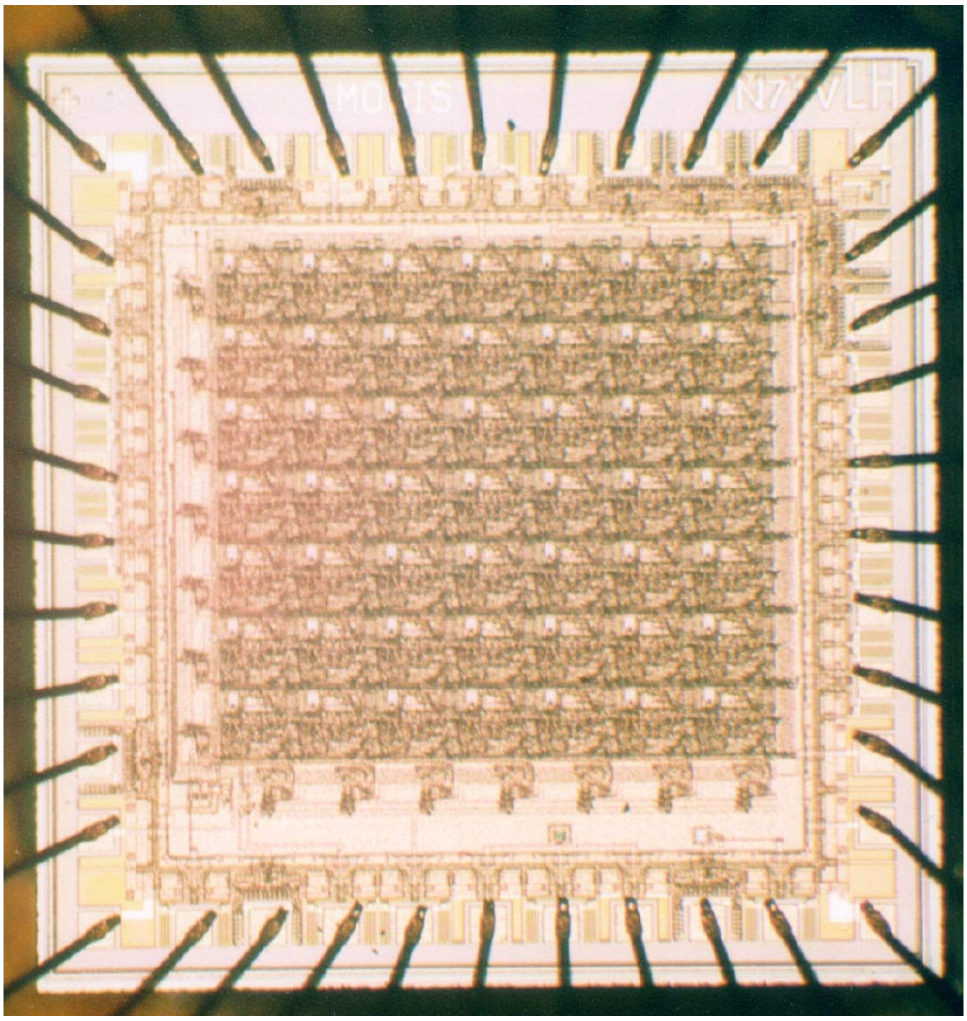


Figure 4.22: Micro photograph of the Gradient2d sensor.

Chapter 5

FTC Sensor: 2-D Direction-of-Motion

5.1 Introduction

For all motion detection schemes some sort of delay or storage element is required. In computer vision block matching schemes for computing the optical flow at least two image frames of different times are compared, see for example [Boh94]. In computer vision gradient schemes for optical flow computation [BFB92], [VGT90] at least two frames are required for the temporal derivative. Continuous time hardware implementations which require a temporal derivative for motion detection include at least one capacitor in the temporal derivative circuit, see for example [DM91], [Hor97] and chapter 3 and 4 in this work. Hardware implementations of Reichardt correlation schemes require an explicit delay element which always includes at least one capacitor [Del93] [HBB+92]. Finally feature based motion detection implementations again require storage elements for feature extraction, for example the temporal edge detector (TED) in [KSK97] is just a non linear temporal derivative circuit.

With only the feature extraction stage, for example the temporal edge detector, velocity cannot be computed. In order to obtain a velocity information from the spike events of the TED, an additional integration stage is necessary, which includes at least one capacitor. For sufficient accuracy this capacitor has to be of considerable size and increases the required chip area per pixel significantly.

The argument is made here that for some applications and high level vision tasks only the direction-of-motion field is required and velocity information is not needed. Small pixel sizes and high density pixel arrays can then be achieved by giving up the functionality of velocity computation, because besides in the feature detection stage no accurate storage element is required. The development of algorithms for high level vision tasks from direction-of-motion fields is still an open field of research. Preliminary results by the author suggest that direction-of-motion fields may very well be used for some problem classes. The focus of this chapter lies on the hardware implementation of a compact sensor for real-time computing of the 2-D direction-of-motion field.

Interestingly the proposed computation of only direction-of-motion has an equivalent in

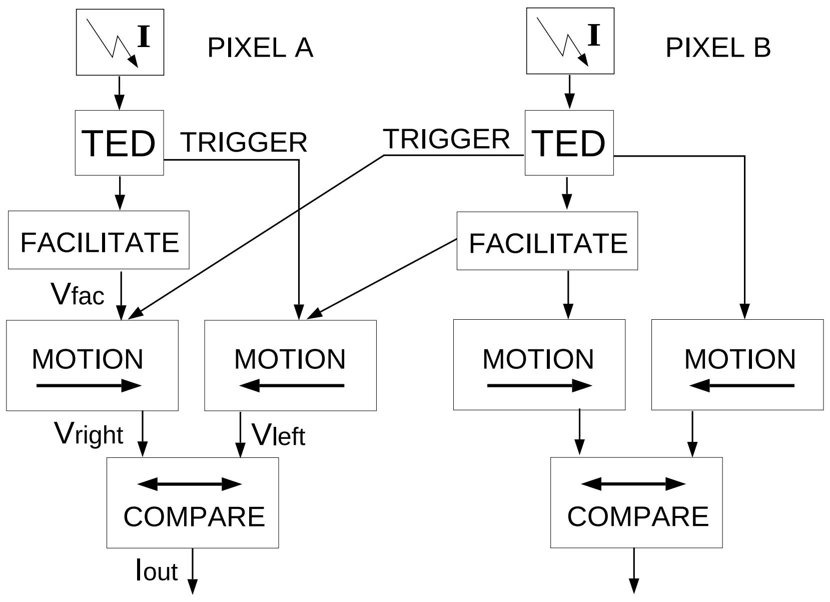


Figure 5.1: Block diagram of two pixels of the FTC sensor. For simplicity the sensor is shown only for motion in one spatial dimension. Main components are the adaptive photoreceptor, the temporal edge detector (TED), a circuit which generates a facilitation signal for a fixed time after an event from the temporal edge detector, a left-to-right and a right-to-left motion indicator each and a comparator. As can be seen only two wires are exchanged between each pair of pixels. This is also true for the algorithm in two dimensions.

biology. Neurons which respond selectively to a particular direction-of-motion (direction-selective cells) are found very early in the visual processing pathway of a wide range of organisms. These neurons are thought to be a key building block in the processing of visual motion. In primates, direction-selective cells are first found in primary visual cortex [Nak85]. Cells which are direction-selective and speed tuned over a wide range of spatial frequencies are first found in area MT [ME83].

Parts of this chapter on the FTC sensor have already been reported in [DHK97a] and [DHK97b].

5.2 Facilitate, Trigger and Compare (FTC) Algorithm

A new direction-of-motion algorithm was developed which met the constraints for a successful and dense implementation in silicon hardware:

1. The algorithm has to be implementable on a discrete square grid. Although hexagonal grids and other symmetries have been used for other implementations, the layout of a square grid is the most straightforward.

2. Only nearest neighbor interaction with at most two communication channels between each pair of pixels must be required. Other than in machine vision where arbitrary addressing is possible, parallel implementations with non multiplexed direct communication schemes that

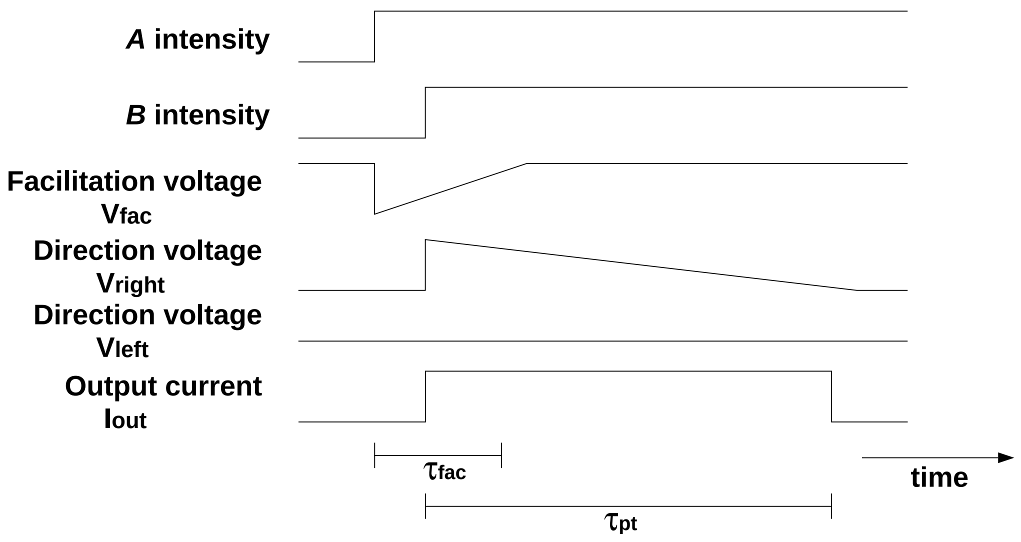


Figure 5.2: Signals within the FTC sensor (idealized). Top traces are the intensities seen by pixels A and B. The facilitation signal V_{fac} is low-active for the time τ_{fac}, the signals from both motion indicators are high-active for the time τ_{pt}. Finally the output current can assume three states: positive for rightward motion, negative for leftward motion and zero for no motion.

require direct communication across more than nearest neighbors are impractical for the cost of wiring.

3. The algorithm must be fault tolerant, because an ideal input from the feature detector cannot be expected.

4. Even though for any motion scheme states have to be stored, the algorithm must not depend critically on the precise value of that state. This allows for the use of small and imprecise storage elements (small capacitors).

5. Several possibilities exist for the representation of the direction-of-motion output. The output has to be readable off the chip easily, but must not require a reset clock.

The idea of the FTC algorithm is to use to occurrence of a feature at one place on the sensor to facilitate its possible motion in all directions, and then trigger the correct direction-of-motion by detecting the same feature at a neighboring place on the sensor at a later time. The features to which the FTC sensor is made sensitive to are contrast edges. A temporal edge detector generates an event if a contrast edge of sufficiently large contrast and velocity moves over it. Both dark-to-bright and bright-to-dark edges are detected. In the following the FTC algorithm is first described for direction-of-motion in one dimension (see Figures 5.1 and 5.2). The extension for detection of 2-D direction-of-motion is explained thereafter.

If a sufficiently strong edge moves from left to right first in pixel A a temporal change of light intensity is detected and the temporal edge detector generates an event in form of a short pulse.

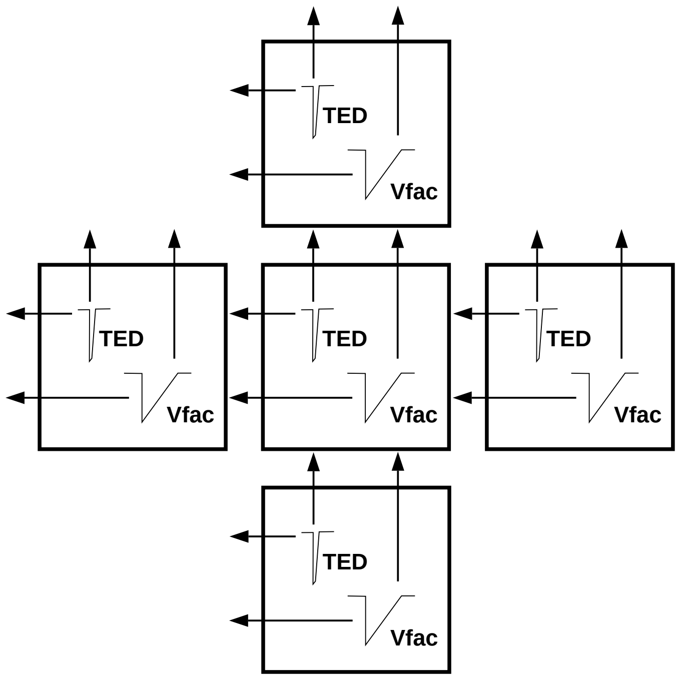


Figure 5.3: 2-D block diagram with five pixels out of the FTC sensor. Every pixel transmits to its upper and left neighbor the temporal edge detector pulse and the facilitation signal. Accordingly every pixel receives from its right and lower neighbor these signals which are used to locally compute the direction of motion. Thus the communication involves only two wires between each pair of pixels.

With this pulse in the facilitation circuit a facilitation signal is started (V_{fac} goes low) and lasts for the fixed time τ_{fac}. As the edge crosses pixel B within this time period τ_{fac}, in pixel A the rightward motion indicator can be triggered (V_{right} goes high). This is because this indicator has been facilitated before. The facilitation signal in A and the TED signal in B are simply ANDed. In pixel A rightward motion indicator and leftward motion indicator, which is smaller or zero, are compared in the comparator and a positive signal I_{out} is generated. If the edge is slow and crosses pixel B after the time τ_{fac} the facilitation in A is over and no output in A will be generated. The null-direction in A (leftward motion) is suppressed, because for rightward motion this indicator receives the TED pulse before it is facilitated.

The extension of the above described 1-D algorithm in 2-D was done by using two independent orthogonal 1-D motion detectors at each pixel. The 1-D output was a tri-state current, which could be negative, positive or zero. In 2-D accordingly two currents of this type are generated, one coding the X direction, the other coding the Y direction. Two independent tri-state currents would be able to generate nine different output states, corresponding to eight directions in steps of 45° and zero. For the 2-D FTC algorithm these currents are *not* independent, though. For any moving stimulus that covers at least the four-neighborhood of a given pixel, the 1-D direction-of-motion sensors for *both* directions will be set off. This means that the zero states of both 1-D detectors are not independent, thus the amount of actually possible direction-of-motion states reduces to five: the four diagonal directions in steps of 90° and the zero output. The 2-D FTC algorithm divides the 2-D stimulus velocity space into four quadrants, and each quadrant is represented by one output state. In effect the sensor output differs at most $\pm 45^o$ from the true direction-of-motion for any stimulus orientation. As will be shown later, due to circuit limitations for some stimulus direction-of-motions in fact the sensor output can also be pointing off diagonal in the four cardinal directions.

In Figure 5.3 the communication scheme between the 2-D pixels is shown. Every pixel transmits its facilitation state and its TED spike to the upper and left neighbors, the facilitation state and TED spike from the lower and right neighbors is received. Therefore only two wires are required between every pair of pixels.

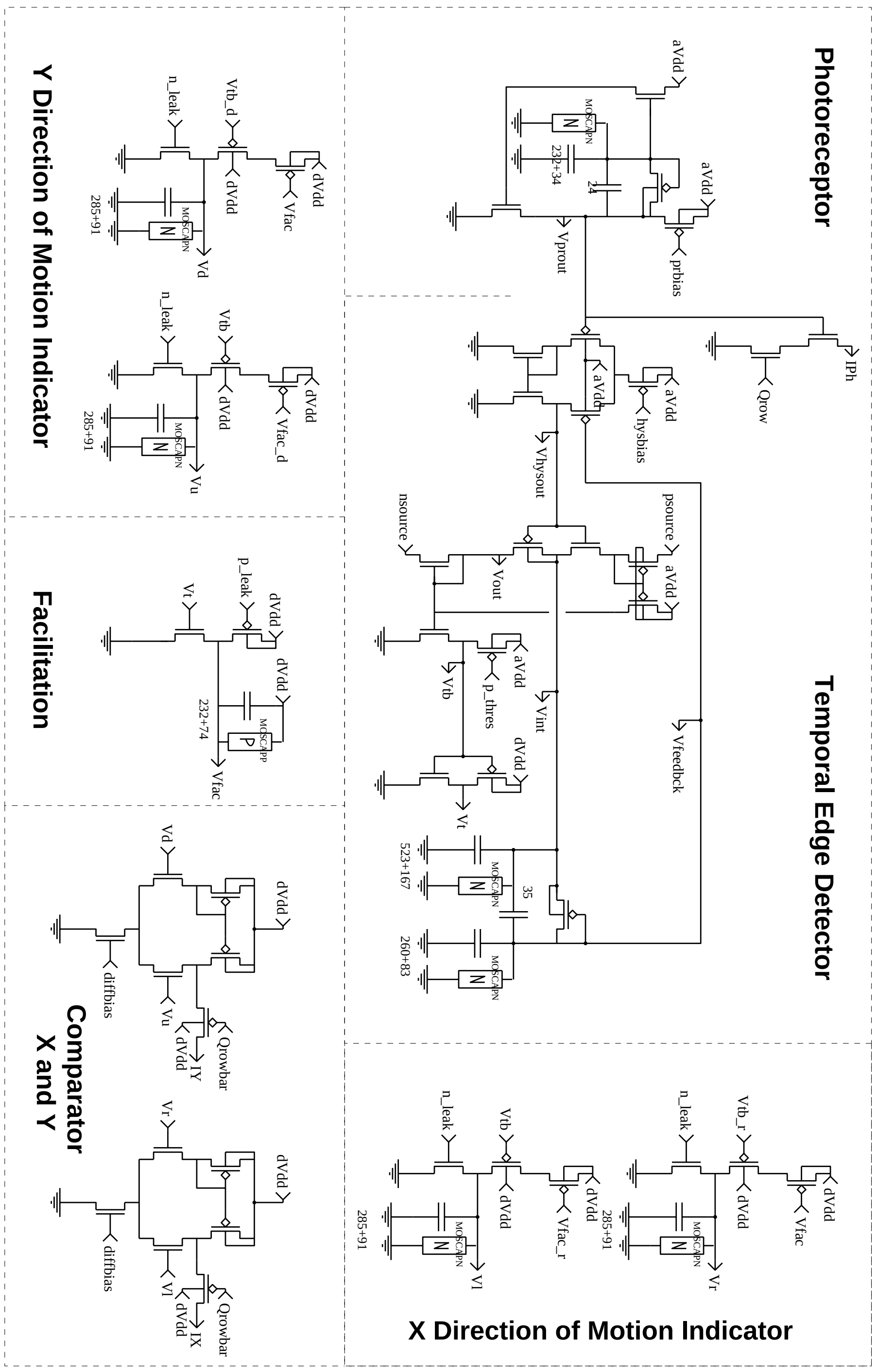


Figure 5.4: Schematic of the FTC pixel.

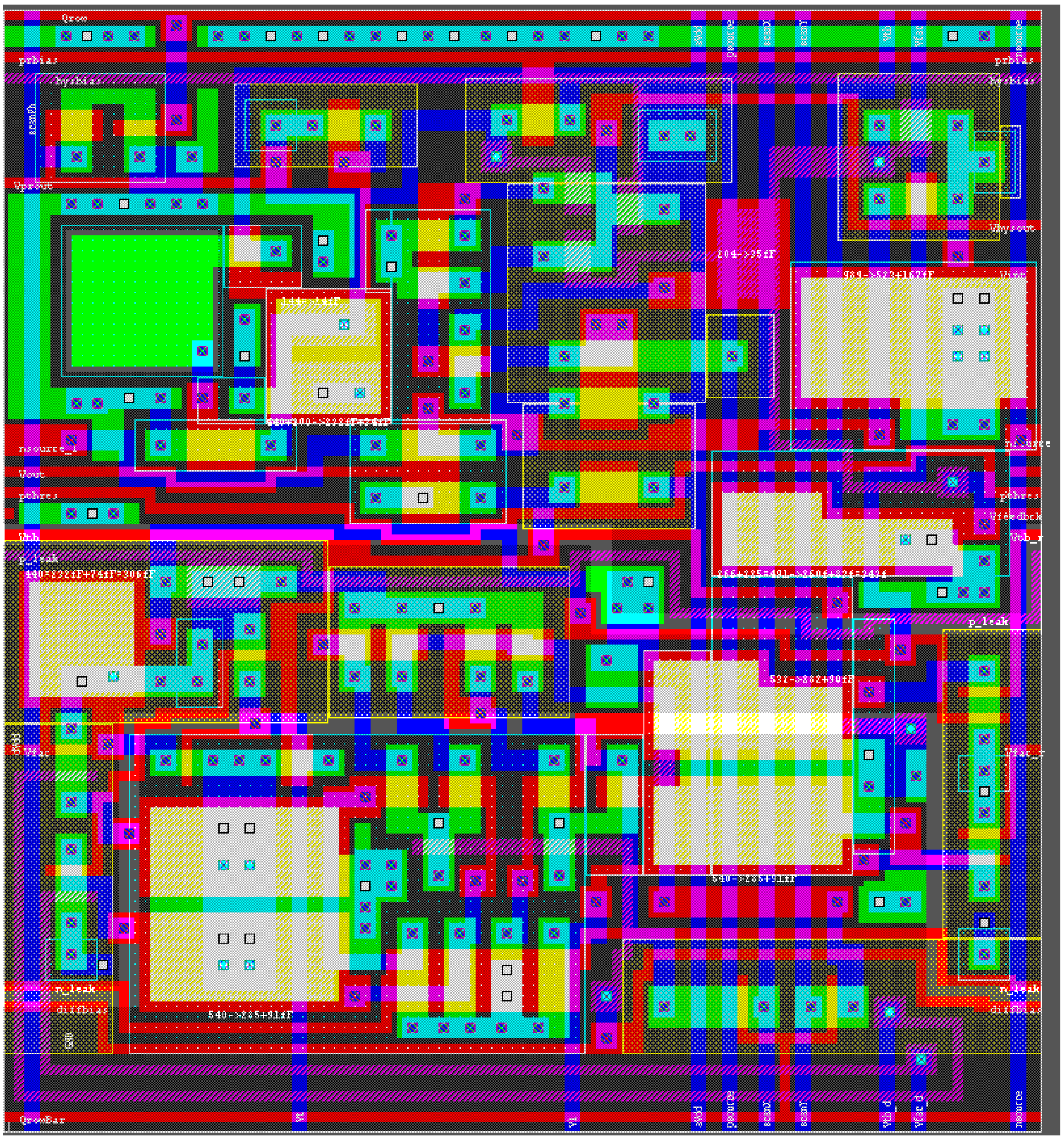


Figure 5.5: Layout of the FTC pixel. The photoreceptor is in the upper left quadrant, the temporal edge detector is located in the upper right quadrant, the lower half of the pixel is occupied by the direction-of-motion circuitry.

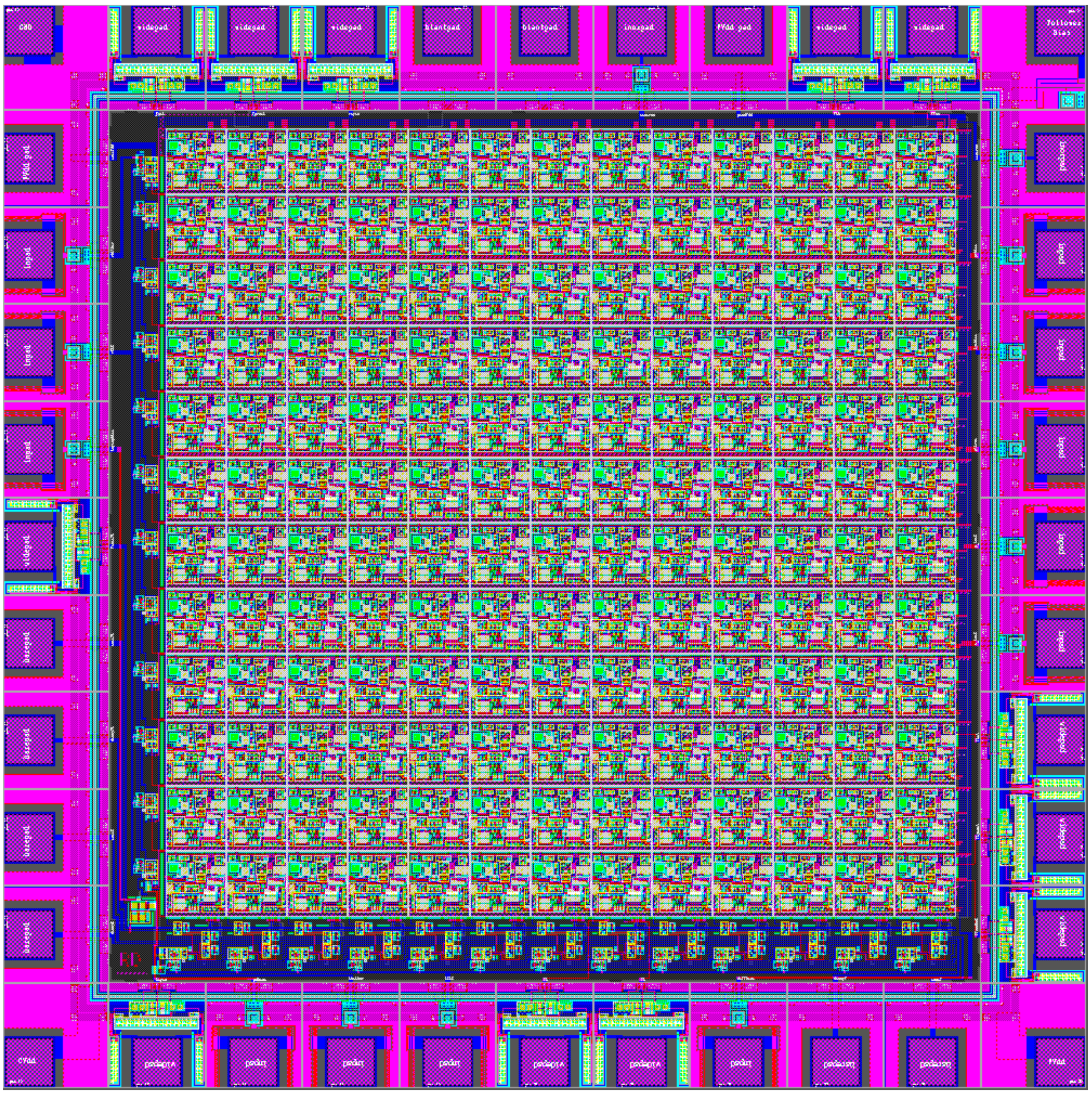


Figure 5.6: Layout of the FTC sensor. The 13 × 12 pixel array fits tightly between the 40 pads. To the left and on the bottom the vertical and horizontal scanners are located.

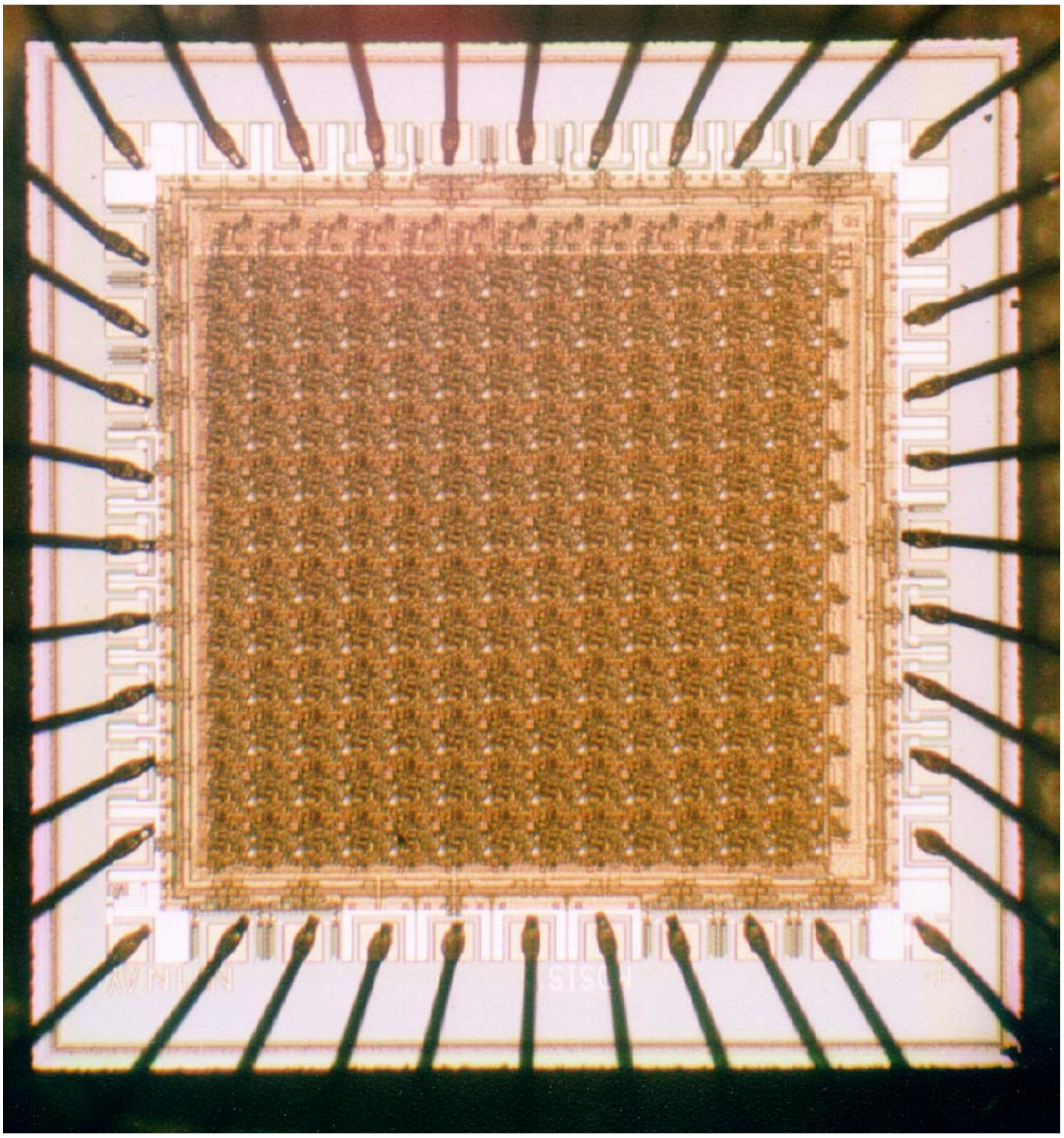


Figure 5.7: Micro photograph of the FTC pixel.

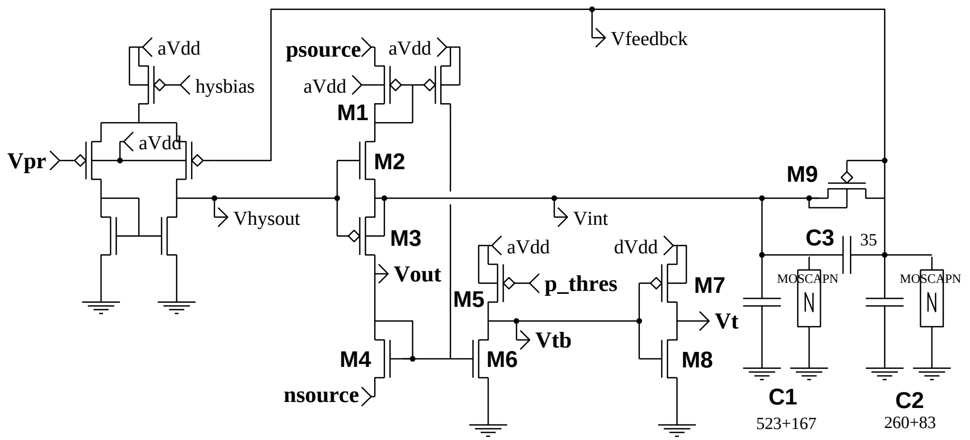


Figure 5.8: Temporal edge detector and thresholding circuit. In response to a change in the photoreceptor voltage V_{pr} the TED output voltages V_{tb} and V_t spike. For a circuit description see the text.

5.3 Circuit and Layout

In the following the building blocks of the FTC sensor, indicated in the pixel schematic in Figure 5.4, are described. These building blocks are implemented in the pixel layout as shown in Figure 5.5. The pixel in turn is arranged as an 13×12 array which together with the scanners and the pads makes up for the sensor chip, as shown in Figure 5.6. A major justification for building a direction-of-motion sensor was to achieve smaller pixel sizes than any other direction selective sensor. Therefore every sensible attempt was made to decrease the size of the circuits. Small devices operate less reliable, but the algorithm was designed to be robust enough to cope with that problem.

Adaptive photoreceptor

The adaptive photoreceptor introduced in [DM94] has some desirable characteristics, most importantly the adaptation to the mean light level. The large dynamic range and low DC offset change with change of ambient light level make this receptor attractive. For dense 2-D layout the use of the receptor is less favorable, because it requires two explicit capacitors, one of which is considerably large and consumes about half of the area of the entire photoreceptor circuitry. For the FTC sensor photoreceptor after circuit software simulations, two chances were taken. First the capacitance of both capacitors was decreased from usually 1 pF to 264 fF and from usually 100 fF to 24 fF. Second, unprecedented in any publication, a MOS capacitor was used for the larger of the two capacitors, the smaller one was stacked on top. This way the capacitors consume less than one seventh of the photoreceptor area.

Temporal edge detector (TED)

Functionally the TED is to detect and indicate fast changes in light intensity, i.e. the TED responds to temporal contrast (see section 2.6.4). For the FTC sensor the necessary TED output is a short digital pulse in response to a temporal contrast change. A previous TED design has very successfully been used in the FS velocity sensor (see section 2.5 and [KSK97]). In the FS sensor, though, the large slow pulse capacitor is part of the TED, increasing its size drasticly. For the FTI sensor [Kra96] the TED is used without the large slow pulse capacitor, but the TED output is not a digital spike. For the purpose of the FTC sensor the author first introduced a thresholding amplifier to the TED design. Additionally the original TED was changed such that not only dark-to-bright but also bright-to-dark edges could be detected.

Briefly the TED functions as follows (see Figure 5.8). V_{pr} is the photoreceptor voltage, V_{out} is the analog TED output, V_{tb} is the thresholded digital spike and V_t its inverse. $V_{nsource}$ and $V_{psource}$ determine the TED sensitivity to falling and rising edges, V_{thres} sets the threshold for generating a V_{tb} pulse from V_{out}. In steady state $V_{feedbck}$ approximately equals V_{pr}, V_{out} is near $V_{nsource}$, V_{tb} is high and V_t low. For increasing light intensity V_{pr} on the input of a differential amplifier increases, V_{hysout} on the output of the differential amplifier increases drasticly, turning on M2 such that current flows through M1 charging up C1. This current is amplified in the tilted current mirror with $V_{psource}$ and raises the voltage V_{out}. If in effect M5 is turned on stronger than the threshold transistor M6 then V_{tb} drops and V_t slams to Vdd. Through the capacitive divider C2 and C3 and the hysteretic element M9 (‘Tobi element’) $V_{feedback}$ increases and eventually matches V_{pr} such that M2 is shut off, V_{out} drops and V_{tb} and V_t are reset to their steady state.

Similarly for falling light intensity V_{pr} drops, $V_{hysdiff}$ drops, current from C1 flows through M4 and raises V_{out} which can, if strong enough, cause a spike for V_{tb} and V_t. The reset again is through the capacitive divider C2, C3 and M9 with $V_{feedback}$ being drawn down to match V_{pr}. The novelty in the circuit for detecting both dark-to-bright and bright-to-dark edges was to connect the output of current mirror M1 to the gate of the second current mirror M4. As a consequence $V_{psource}$ amplifies dark-to-bright edges, but $V_{nsource}$ amplifies both edges.

For the layout of the TED similar chances as in the photoreceptor were taken. Instead of using for C1 a 1.5 pF polyI-polyII capacitor as in previous designs, a 690fF MOS capacitor was used. Also for C2 a MOScap was used, its size was 343 fF, the capacitive divider ratio C2/C3 was set to 10.

As can be seen in the circuit schematic, analog and digital Vdd (aVdd, dVdd) are kept separate even within the pixel. Although this strategy causes some major difficulties in the layout (one more wire has to be routed to the appropriate location within the pixel and two well potentials for the pFETs are to be kept separate), it can be crucial for the proper operation of the edge detector. The switching noise introduced by the digital circuitry to dVdd could cause false TED spikes, if dVdd were used for the TED.

Facilitation circuit and direction-of-motion indicator

The facilitation circuit is triggered by a TED spike and holds the facilitation state for the time τ_{fac}. The direction-of-motion indicators are triggered if a TED spike occurs during the facilitated state, they indicate direction-of-motion up, down, left or right.

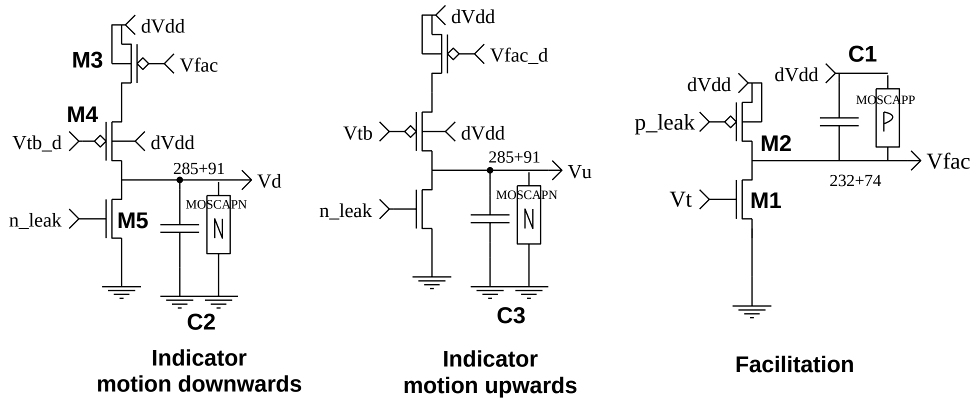


Figure 5.9: Facilitation circuit and direction-of-motion indicator. Only the indicators for the Y direction are shown. There is an equivalent pair for the X direction.

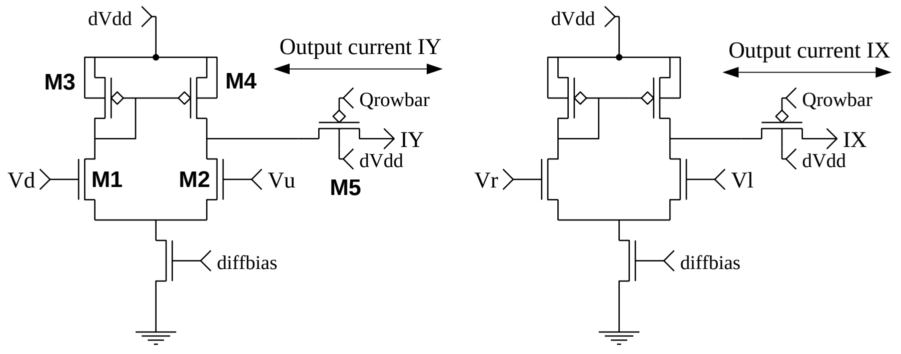


Figure 5.10: Direction-of-motion comparator. The direction of motion indicator outputs V_u, V_d and V_l, V_r are compared and signed currents I_x and I_y are generated.

In the facilitation circuit M1, M2 and C1 the facilitation capacitor C1 is charged to GND through M1 on the occurrence of a TED spike. Through the leakage transistor M3 the active low facilitation voltage V_{fac} on C1 then leaks linearly back to Vdd.

The motion indicator circuit for example for downward motion is facilitated with V_{fac} on M3. If during the time when $V_{fac} < Vdd - 0.7\,V$ the lower neighbor sends a TED spike V_{tbd}, the indicator capacitor C2 is charged to Vdd and the indicator voltage for downward motion V_d goes high. The leakage voltage V_{nleak} on M5 sets the persistence time τ_{pt} of the output. The motion indicator circuit for upward motion receives the facilitation signal from the lower neighbor and V_u goes high if a TED spike occurs in time. The indicator circuits for left and rightward motion interact similarly with the neighbor to the right.

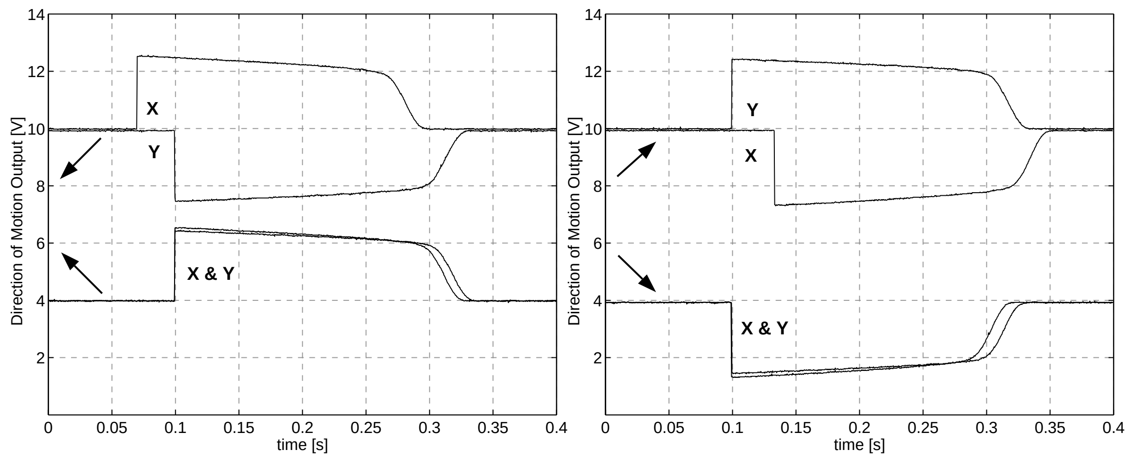


Figure 5.11: Single pixel motion output. On the top traces of the left figure the X and Y motion output for stimulus direction in 225^o is shown. The Y component is activated one inter pixel travel time after the X component. The lower traces show the case for the stimulus direction 135^o. Both components are activated at the same time. There is no transient in the beginning of the output, but there is an off transient because of the finite time the comparator needs to shut off the output. The different persistence times are due to device mismatch. The right figure shows the motion output for the two other stimulus directions.

Comparator

Two comparators receive their input from the direction-of-motion indicators and generate a signed current depending on which direction-of-motion has been detected more recently. If no motion was detected, i.e. all indicator voltages are low, zero current is generated.

More specifically a comparator consists of a differential pair and a current mirror, see Figure 5.10. If V_d is larger than V_u, meaning that downward motion has been detected more recently than upward motion, then the current through M1 is larger than the current through M2. By virtue of the current mirror M3 and M4 in this case a positive output current I_y is sourced from the comparator and can be read out through the digital transistor M5. For the case $V_d < V_u$ current is drained into the comparator and a negative I_y can be read out. The comparator for the X direction works equivalently for the inputs V_l and V_r. In most cases one of the indicators will be zero if the other indicator is activated. The current generated by the comparator then is constant for the time τ_{pt} and decays very quickly to zero after that time, leaving almost no time where the output is ambiguously small.

5.4 Results: Sample outputs

Before flow fields generated by the whole sensor are shown, in this section the direction-of-motion output of a single pixel is demonstrated. From the previous discussion of the FTC algorithm it is expected, that for stimuli moving in the direction of one quadrant the direction-of-motion output points diagonally in this quadrant. This output would be constant during the persistence time τ_{pt}

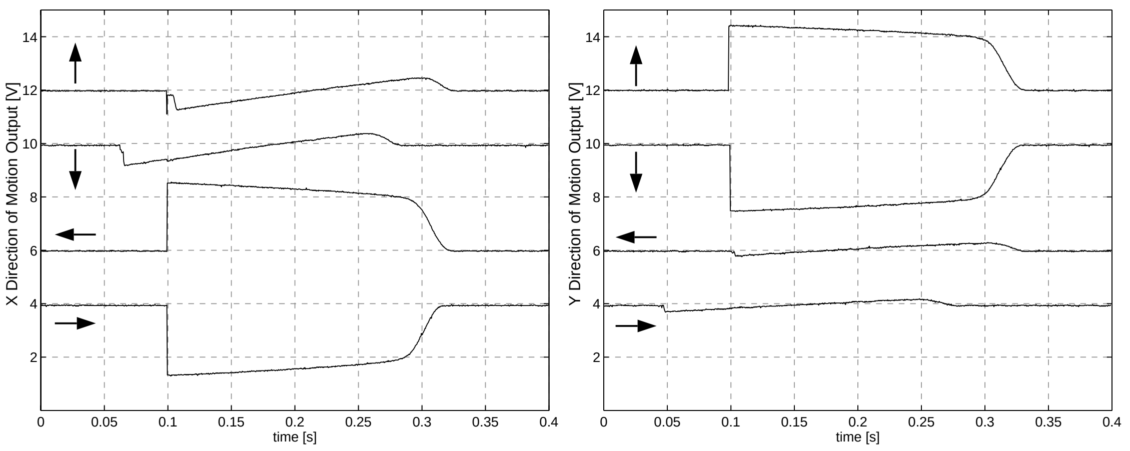


Figure 5.12: Single pixel motion output for stimulus motion in cardinal directions. On the left the single pixel motion output for X direction is shown. For stimulus motion in exact Y direction no motion in X direction is reported. Similarly for stimulus motion in exact X direction no motion in Y direction is reported. The slight deviation from zero for the null direction components is caused by different leakage times τ_{pt} due to device mismatch.

and would vanish thereafter. In Figure 5.11 the actual motion output of one pixel is shown. A bar was moved across the array in the four diagonal directions, and both components of the direction-of-motion vectors were recorded over time. Only the relative sign of X and Y components are significant.

For the stimulus direction 135^o both components go high at the same time when the edge has crossed the two orthogonal pairs of pixels, as explained before. Both outputs stay high for almost the same period of time. No transient at the onset is observed, but the offset of the motion signal lasts about 25 ms. Additionally both components decay to the zero state at slightly different times. This is due to device mismatch. For stimulus movement in 225^o X and Y component are activated at different times, the X component is determined earlier than the Y component. This effect is intrinsic to the FTC algorithm and will be discussed later. This transient time at the onset of the motion vector lasts for one inter pixel travel time. Similarly for stimulus movement in 45^o both components are determined at different times, in this case the Y component has an earlier onset. Finally in 315^o again both components are almost identical, except for a slightly different persistence time.

In Figure 5.12 the special case of stimulus movement in one of the four cardinal directions is shown. In this case, unlike the theoretical expectations from the FTC algorithm, in fact one of the components remains zero. Thus the total number of output states together with the four states just described becomes eight, which is a desirable feature. On the left side of the figure the X component is shown for all four stimulus directions. For 90^o and 270^o it remains zero, for 0^o and 180^o a negative and positive current is reported, respectively. Similarly for the Y component a zero output current is generated for 0^o and 180^o. The reason for the fact, that zero output can be generated in a 2-D implementation of the FTC algorithm is given in later in section 5.10.1.

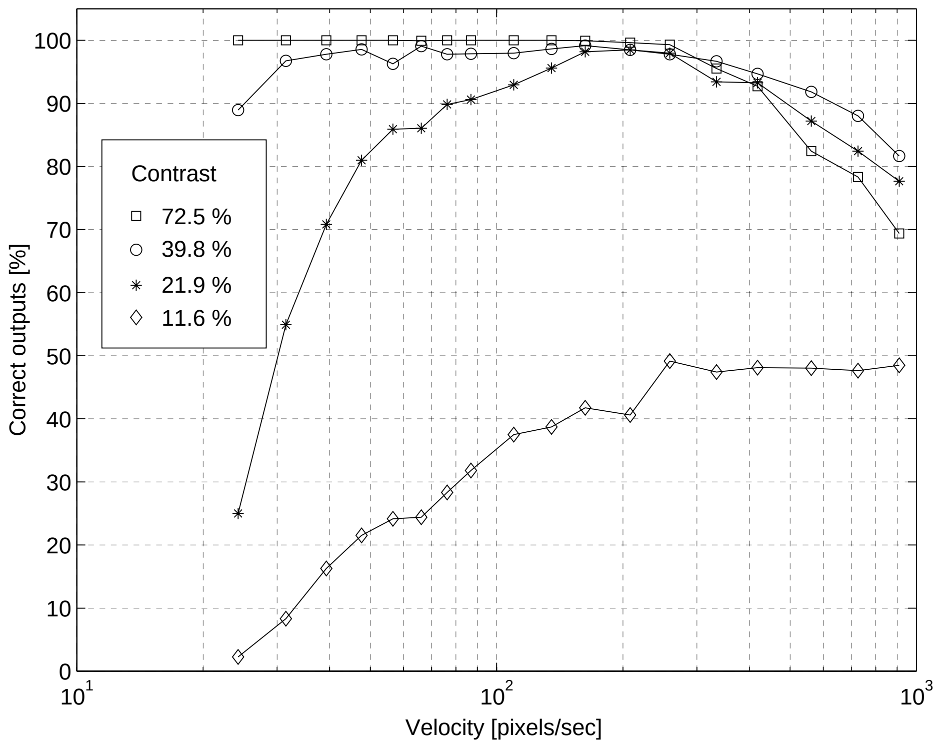


Figure 5.13: FTC contrast and velocity sensitivity. For contrasts above 20% the FTC sensor reports the correct direction-of-motion in 80% of the cases for a velocity range of more than one order of magnitude. For contrasts above 40% the direction-of-motion output is computed correctly for velocities from 25 pixels/sec up to 450 pixels/sec.

5.5 Results: Velocity and Contrast Sensitivity

In this section the performance of the elementary motion detector of the FTC sensor is determined. Ideally the sensor should report the direction-of-motion of the stimulus independently of the stimulus velocity and contrast. In reality the range of optimal operation is limited. In order to asses the reliability of the elementary motion detector of the FTC sensor it is not sufficient to only test one pixel of the array. Fluctuations over the array would not be captured. For the experiments in this section a gray bar on white background was moved over the array with the ‘can’ stimulus device. The output of the whole array was recorded over multiple stimulus presentations, saved to a file and processed in Matlab. For a given stimulus orientation, velocity and contrast the vectors were counted which pointed in the stimulus direction $\pm 22.5^o$. With this rating scheme chance is at $45/360 \cdot 100\% = 12.5\%$. The right and lower boundary pixels were not counted because due to the missing neighbor they cannot generate a two dimensional output. Figure 5.13 shows the result for different contrasts of the edges and for a velocity range of over one and a half orders of magnitude, the stimulus orientation of 135^o was kept constant.

For high contrast edges the reported direction-of-motion was 100% correct from the lowest velocity that could be generated up to 110 pixels/sec. This means that for 10 consecutive stimulus presentations every one of the $11 \times 12 = 132$ pixels reported correctly, which is remarkable. For

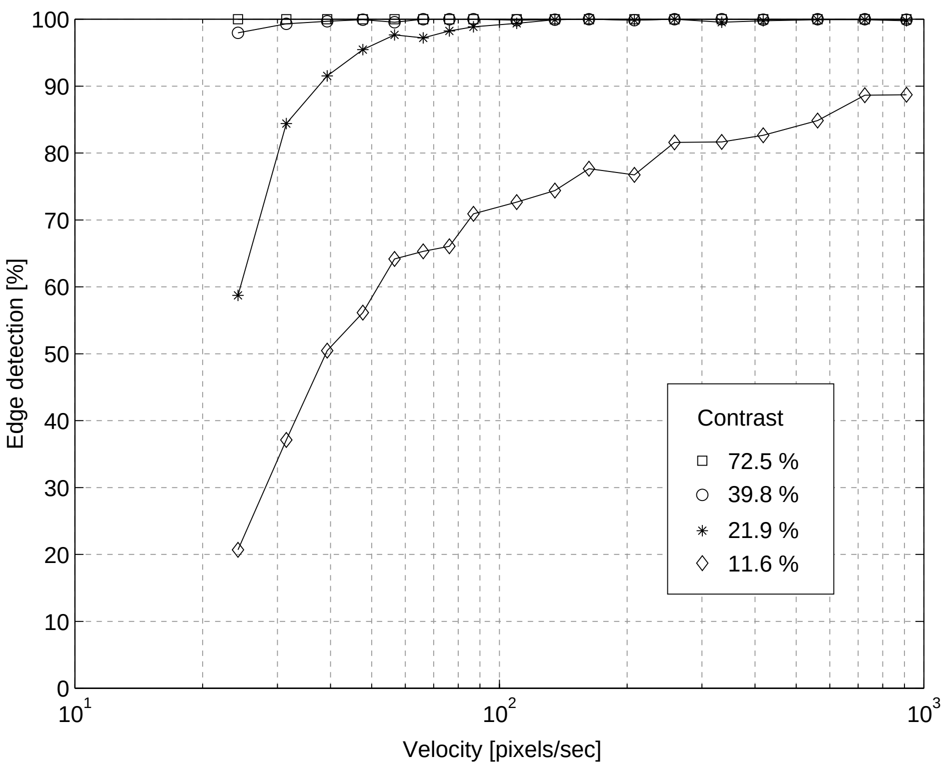


Figure 5.14: Reliability of the temporal edge detector. For edges of different contrast the percentage of pixels is plotted that generate a TED spike in response to edge presentation. It can be seen, that the spike detection becomes better for higher contrasts and higher speeds. The temporal edge detector is approximately responding to temporal contrast.

higher velocities the reliability is less than 100%. For contrasts as low as 20% the performance remains satisfactory ($> 80\%$) for over one order of magnitude of velocity. Even for a very low contrast edge of 11.6% the sensor output is still well above chance.

Another very interesting finding of this experiment is the dependency of the sensor reliability on the stimulus velocity. As expected, for low velocities the high contrast edges give better results, and the performance increases for increasing velocity because of the increasing temporal contrast. For high velocities, though, the performance for high contrast edges degrades more than for low contrast edges. The reason for this phenomenon is that for very high temporal contrasts and a fixed TED threshold multiple spikes can be generated for one edge, which can lead to wrong motion outputs. This will be discussed in section 5.10. Figure 5.14 proves that in fact the temporal edge detector spike generation probability increases with temporal contrast. This plot is generated from the same data as Figure 5.13. It does not show, though, if one or multiple spikes are generated.

In the previous experiment it was found that for a high contrast bright-to-dark edge the sensor output was 100% correct even for the lowest stimulus velocity that could be generated with the 'can' stimulus device. In order to find out to how low stimulus velocities the FTC sensor can find the correct direction-of-motion, in a different experiment the 'bicycle wheel' stimulus device was used, which can generate velocities that can barely be seen with the eye. As stimulus again a high contrast bar pattern was used. As compared to the previous experiment the facilitation time τ_{fac} and the persistence time τ_{pt} were increased. The surprising result was that the FTC sensor reported with 100% confidence the correct direction-of-motion *down to the velocity of 0.5 pixels/sec*. This means that the edge took 35 seconds to move diagonally across the sensor, which is a very slow motion.

5.6 Results: Orientation tuning curve

It was explained earlier that the 2-D FTC algorithm separates the velocity space into four directions of motion. This was tested in an experiment, where an edge was moved over the sensor with a constant velocity but in different directions. As in the previous experiment not only one pixel was tested but the whole array was recorded over multiple stimulus presentations. In Figure 5.15 the results are summarized. On the Y axis the percentage of positive vector components (upward bars) and negative vector components (downward bars) is plotted for both direction-of-motion vectors over all stimulus orientations. 100% means that in the given stimulus direction all pixels of the array over multiple presentations reported the respective positive or negative vector component. According to the theory it is expected that for example the X component of the output reports a positive value for edge directions between 270^o and 90^o and a negative value for angles from 90^o to 270^o. There would be no angle for which no output was generated. From the experiment it can be seen that this is almost confirmed. The sensor reports either 100% positive or 100% negative vector components. There are stimulus directions, though, in which the sum of the positive and negative vector component outputs is less than 100%. This means that zero output occured. A pixel can report zero motion if two neighbors are seeing the stimulus in close sequence. This is the case for stimulus orientations in the for cardinal directions 0^o, 90^o, 180^o and 270^o. The underlying reason lies in the circuit and will be discussed later. In effect the orientation tuning curve does not look like a perfect square wave but comes closer to a cosine and sine wave for X and Y vector component, respectively.

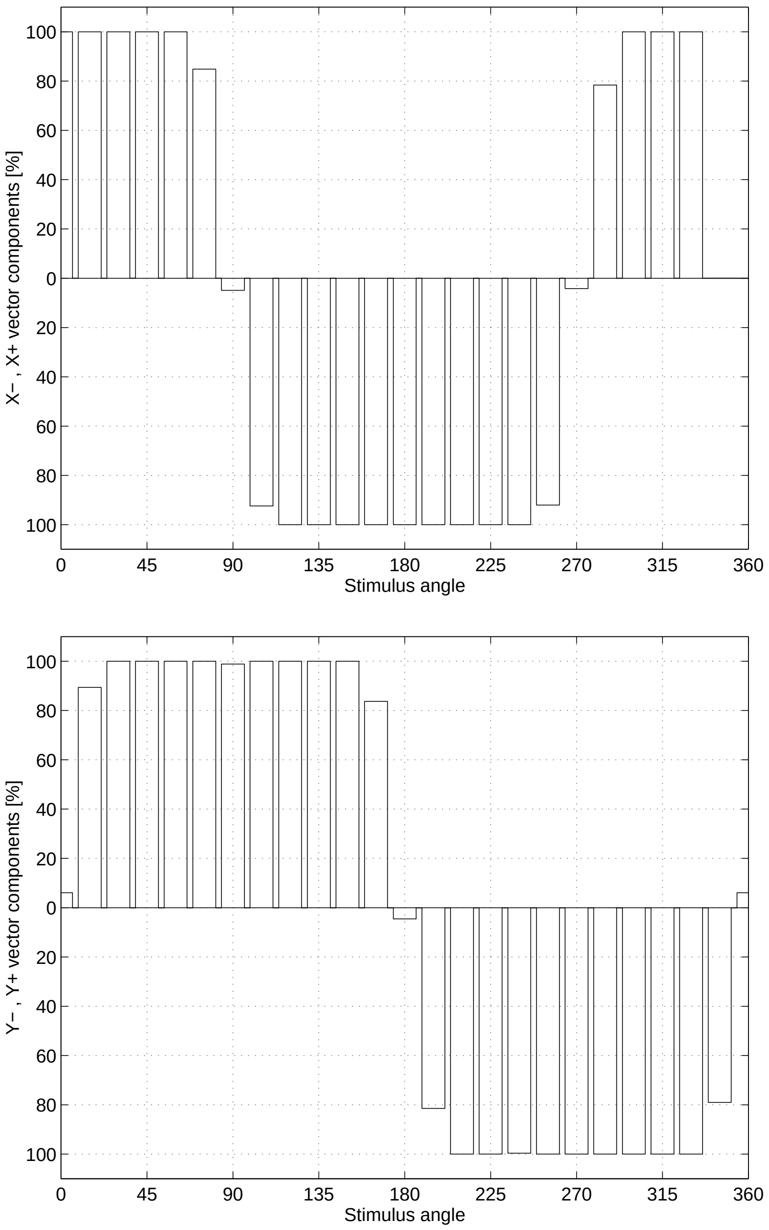


Figure 5.15: Orientation tuning curve. The percentage of pixels is plotted that responds to a certain direction-of-motion for a given stimulus orientation. For example on the top figure a positive X direction is computed for stimuli moving from close to -90^o to close to 90^o. At $\pm 90^o$ some pixels do not report motion in X direction.

5.7 Results: Sample Flow Fields

Some effort was invested to demonstrate the performance of the FTC sensor. A demonstration setup was built and software was written that allowed to scan the vector field and the gray value image off the FTC sensor into a computer and display both on the computer monitor in real time.

The flow fields shown in Figures 5.16 and 5.17 are snapshots of the FTC sensor output. One frame of the vector field was read into the computer and stored to disk. The persistence time τ_{pt} was chosen such that the vectors would persist on the sensor array during stimulus presentation. In the following image sequences the persistence time is set shorter. As stimulus for the first two examples a round object was used, which was either moved towards the sensor from above or away from it. The resulting flow fields are expanding and contracting, respectively. Also shown is a simulation of a direction-of-motion sensor with a continuous angular resolution (see second images). The original snapshot is subtracted from this simulated output and the resulting difference vector field is shown in the bottom figures. It becomes obvious that the best match between continuous and discrete sensor is along the diagonals. The further away from the diagonal direction, the greater the error. In Figure 5.17 a 'wagon wheel' like stimulus was rotated above the sensor. Again the original sensor output is compared with the simulation of a continuous direction-of-motion sensor and the difference is plotted. Clearly along the diagonals the best match is obtained.

By the nature of motion sensors it is difficult to demonstrate their output in a printed medium. To give an impression of not only static images but of the time course of the sensor output for a moving stimulus, software was written to scan the vector field and the gray value image into the computer memory as fast as possible over a certain time. The image sequence was then stored to a disk, the vector field and the gray value image were superimposed and are shown in this work. All image sequences are to be read left to right and top to bottom.

With this method the following frame rate was achieved: The scanner vertical clock frequency (VCLK) was 3.33 kHz as set by the function generator. The horizontal clock (HCLK) frequency, which is generated by the vertical SYNC pulse was measured as 255 Hz, which corresponds to 3.33 kHz divided by the vertical pixel count (12) plus one. The total frame rate is given by the horizontal SYNC pulse and was measured as 18.1 Hz, which corresponds to 256 Hz divided by the horizontal pixel count (13) plus one. For the recording of the following sequences, one frame was taken within 1/18.1 Hz, or 55 ms. Then the time of one frame was needed to store the data, such that the next frame was taken after a 55 ms delay. The resulting effective frame rate shown in the sequences was therefore 1/110 ms=9.1 Hz, i.e. the time between each pair of frames was 110 ms.

The sample flow field sequences are to demonstrate the usefulness of the FTC sensor. Natural stimuli such as fingers and a hand were used, as well as gray value images printed on paper for the rotating stimuli such as spirals and 'wagon wheels'. It is remarkable that even with a resolution of only 13×12 pixels an entire hand can be detected and the correct flow field can be computed. All snapshots and image sequences are shown here as they were read off the chip, the data has not been thresholded, corrected or otherwise changed in any way.

The first sequences were generated with an 8 mm lens. In Figure 5.18 two fingers were moved right down, then stopped, then moved back up again. It can be seen how the vectors are generated on the bright-to-dark edge and how they persist over some fixed time. In the frames where the fingers are stopped, the vectors disappear because no new motion information is available to the sensor. In the following sequence a hand was moved down, up and down again. It can be seen how every single finger causes motion vectors. If the hand stops, the vector field disappears. In the following sequence it is demonstrated the FTC sensor can be used for segmentation from motion tasks. A finger was moved down right, whereas at the same time a thin pen was moved in the opposite direction. In frame six it can be seen how the motion vectors face each other at the leading edge of both objects. The same situation occurs when both objects are moved back. For the next two sequences a 4 mm lens was used to obtain a wider field of view. In Figure 5.24 a dark circle was moved with constant velocity towards the sensor from above. The resulting flow field is expanding. When the object is far from the sensor the apparent velocity on the chip is small, therefore in the first few frames the vector field is sparse. A similar flow field is expected for the case when the sensor were moving towards a wall, which means that the focus of expansion in 2-D could be determined using the FTC sensor. In Figure 5.25 a bright circle on dark background was moved with a constant velocity away from the sensor. As a result a converging vector field is obtained. In the following two sequences the sensor response for a thin rotating spiral is shown for both directions of rotation, then the response for a rotating ‘wagon wheel’ pattern is demonstrated. The axis of rotation, which in this case is outside of the field of view, and the direction of rotation could be determined from the flow field. The same pattern is used for the following sequence, but a 2.6 mm lens was used for a wider field of view. Due to the short persistence time that was chosen to bring out the features of the stimulus, many transient vectors are visible. These transients are discussed later. For applying the FTC sensor a higher pixel resolution were used and the transients would be less dominant. Nevertheless the axis of rotation and the direction of rotation could be determined. In the next two sequences it is demonstrated that both dark-to-bright and bright-to-dark edges can be detected. For the previous flow fields the sensor was made sensitive to only one edge in order to avoid aliasing due to the low resolution and the complexity of the stimuli. In Figure 5.30 a finger is moved downward and then stopped. It can be seen that both edges cause motion vectors as long as the object is moving. In the next sequence the aperture problem is demonstrated by a round dark object moving upward. For the leading edge the apparent motion is opposite to the apparent motion of the trailing edge. The last sequence shows how occlusions affect the flow field. A round object was moved over a finger, which was held stationary in front of this object. As a result the flow field is interrupted at the location of the occluding object.

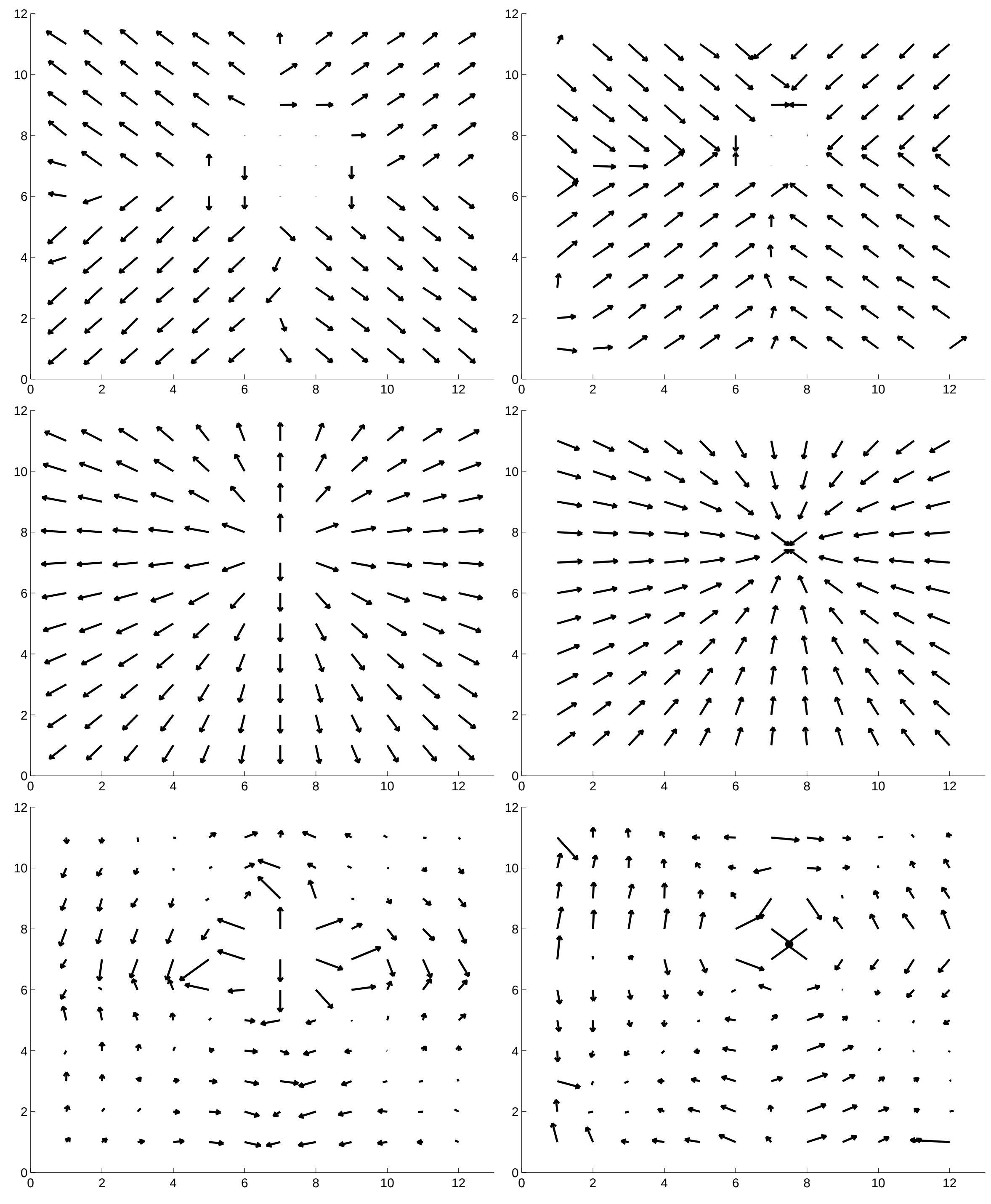


Figure 5.16: FTC flow field and simulated flow field. In the top two figures snapshots of the FTC sensor output are shown. A round object was moved towards the sensor from above (left figure) and away from the sensor (right figure). Even without knowledge of the underlying stimulus used for these sensor snapshots, the foci of expansion and convergence could be determined from the flow fields. The two figures in the middle show the simulation of a direction-of-motion sensor with continous angular resolution. On the bottom figures the difference between the measured flow field and the simulated continous flow field is shown. As expected, the best match is along the diagonals.

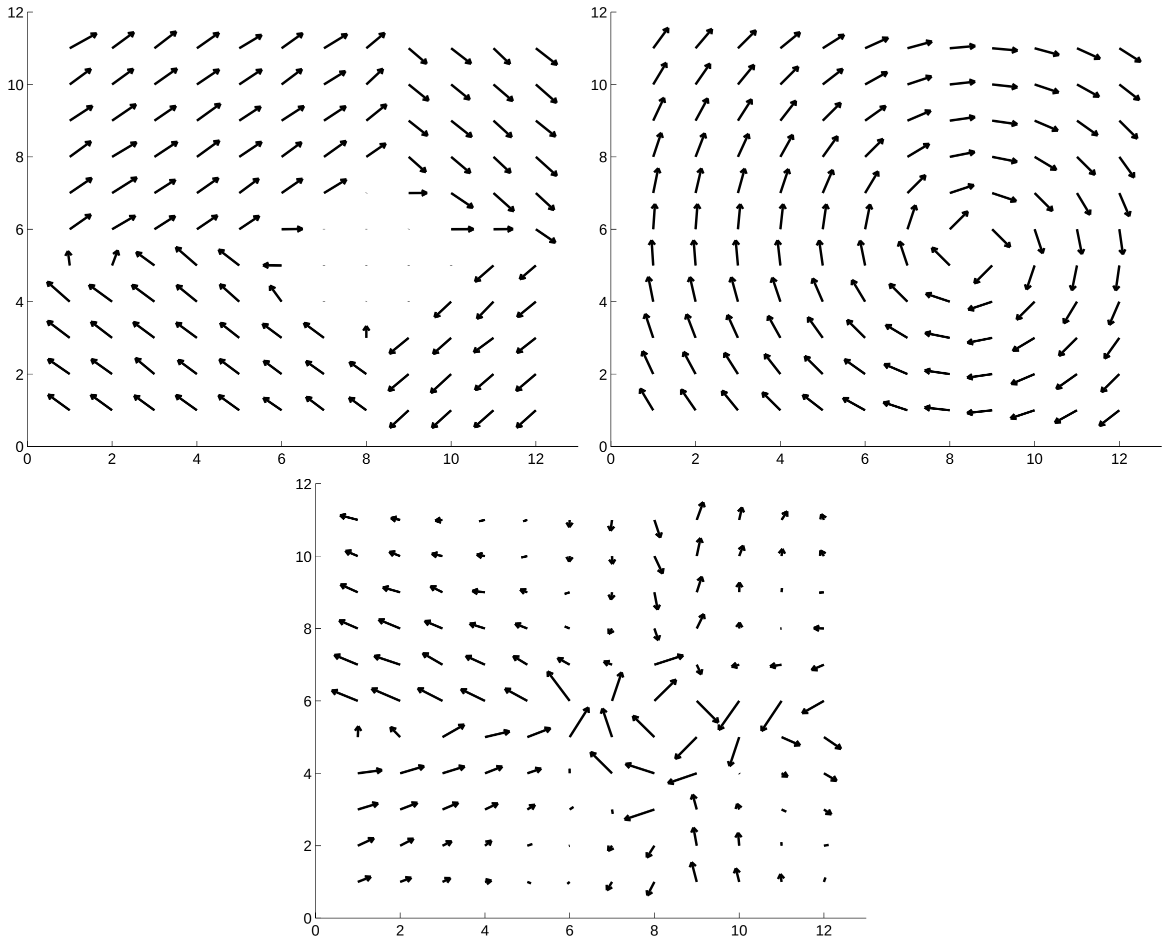


Figure 5.17: FTC flow field and simulated flow field. The snapshot in the left figure shows the FTC sensor output for a rotating stimulus. The axis of rotation and the direction of rotation can be determined. As in the previous figure, the best match between FTC sensor output and a flow field with continuous angles (right figure) is along the diagonals, as shown in the bottom figure.

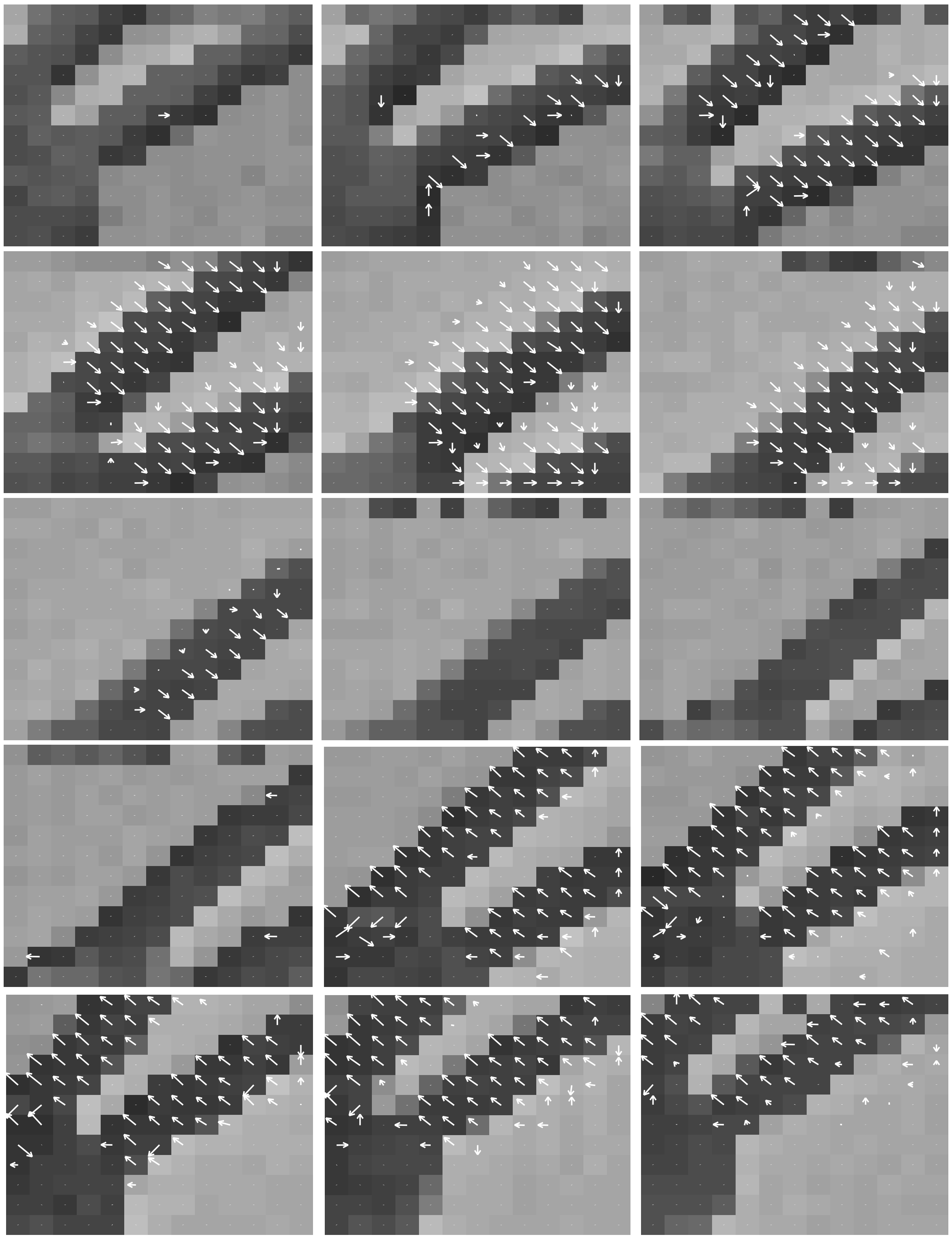


Figure 5.18: FTC flow field: Moving fingers

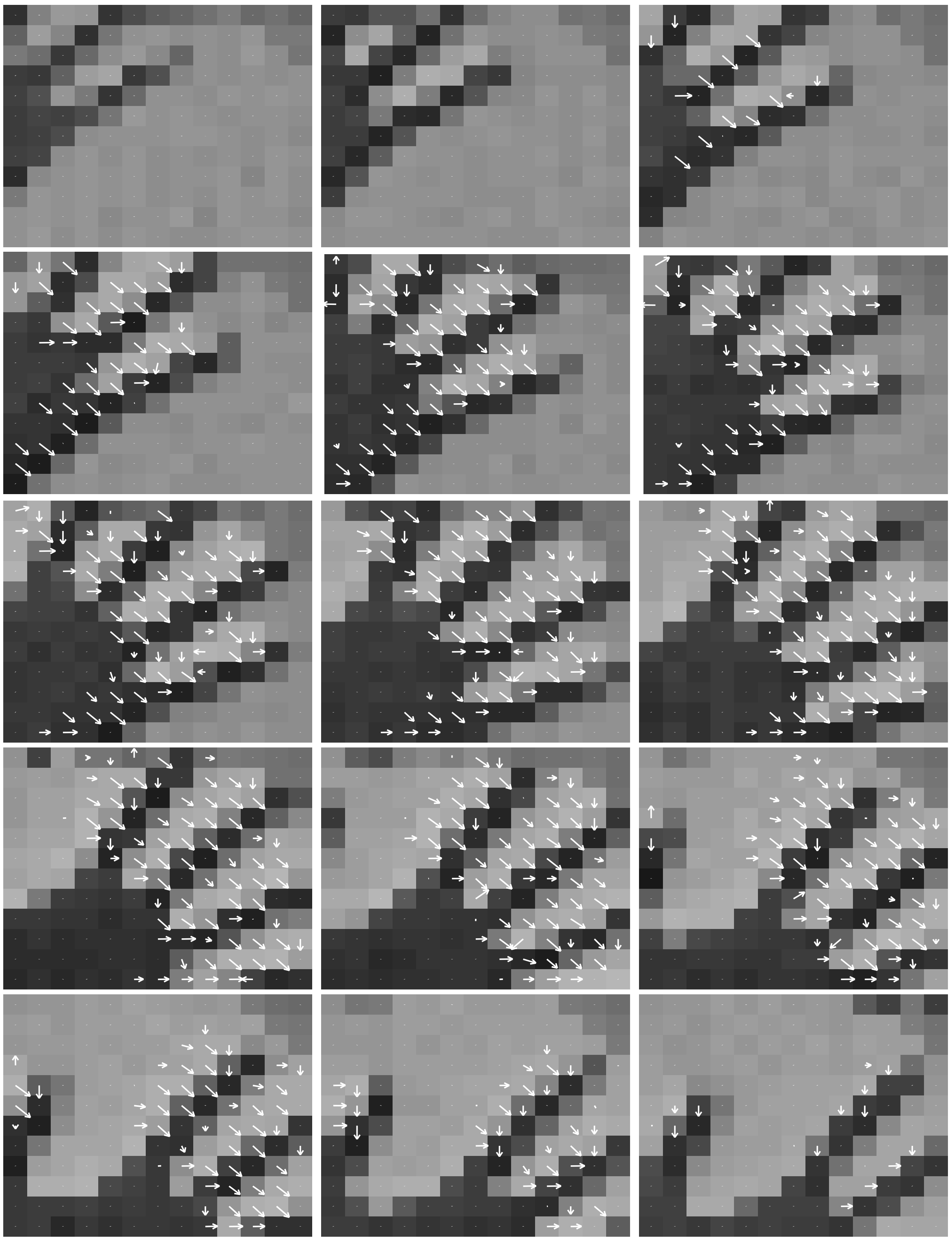


Figure 5.19: FTC flow field: Moving hand.

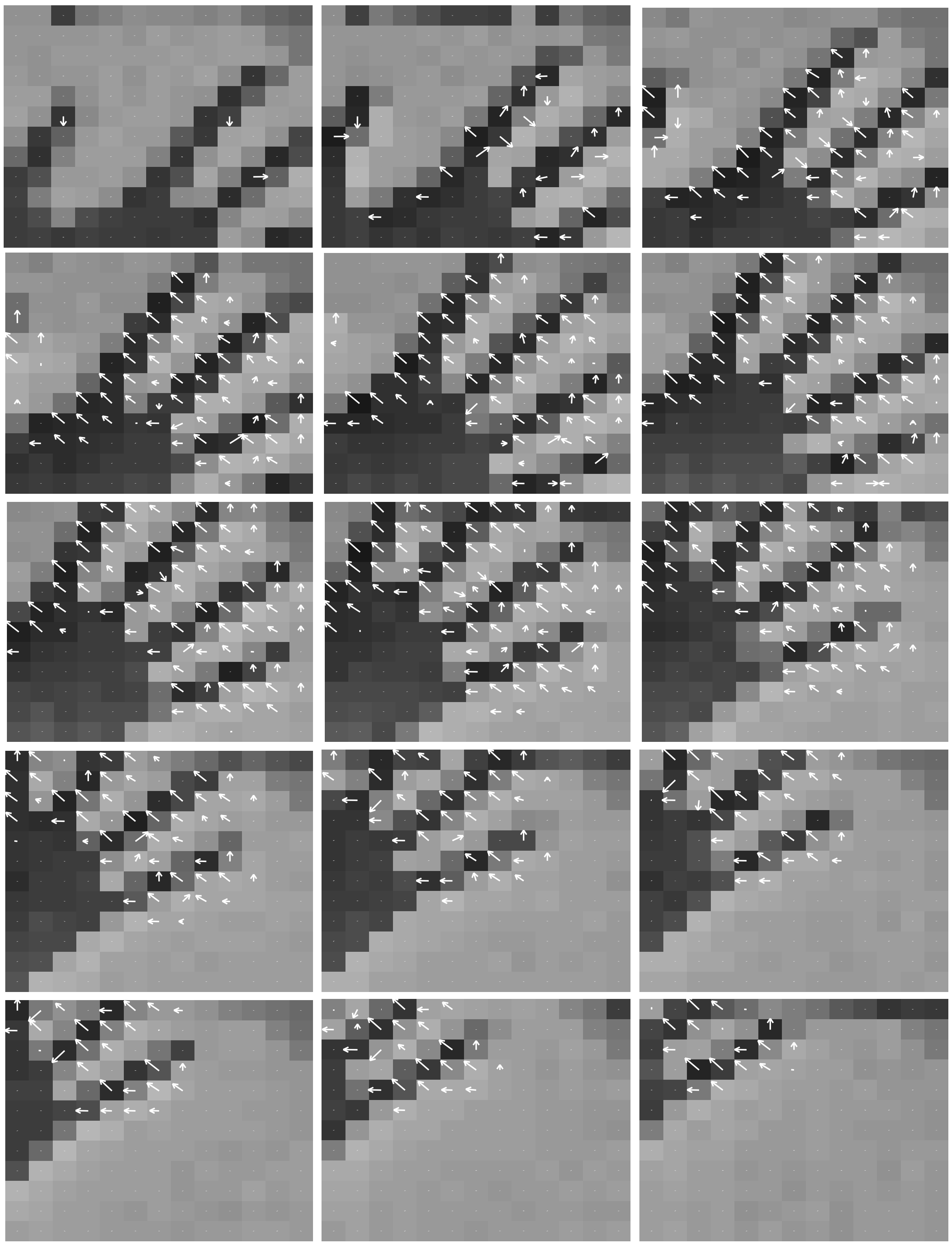


Figure 5.20: FTC flow field: Moving hand.

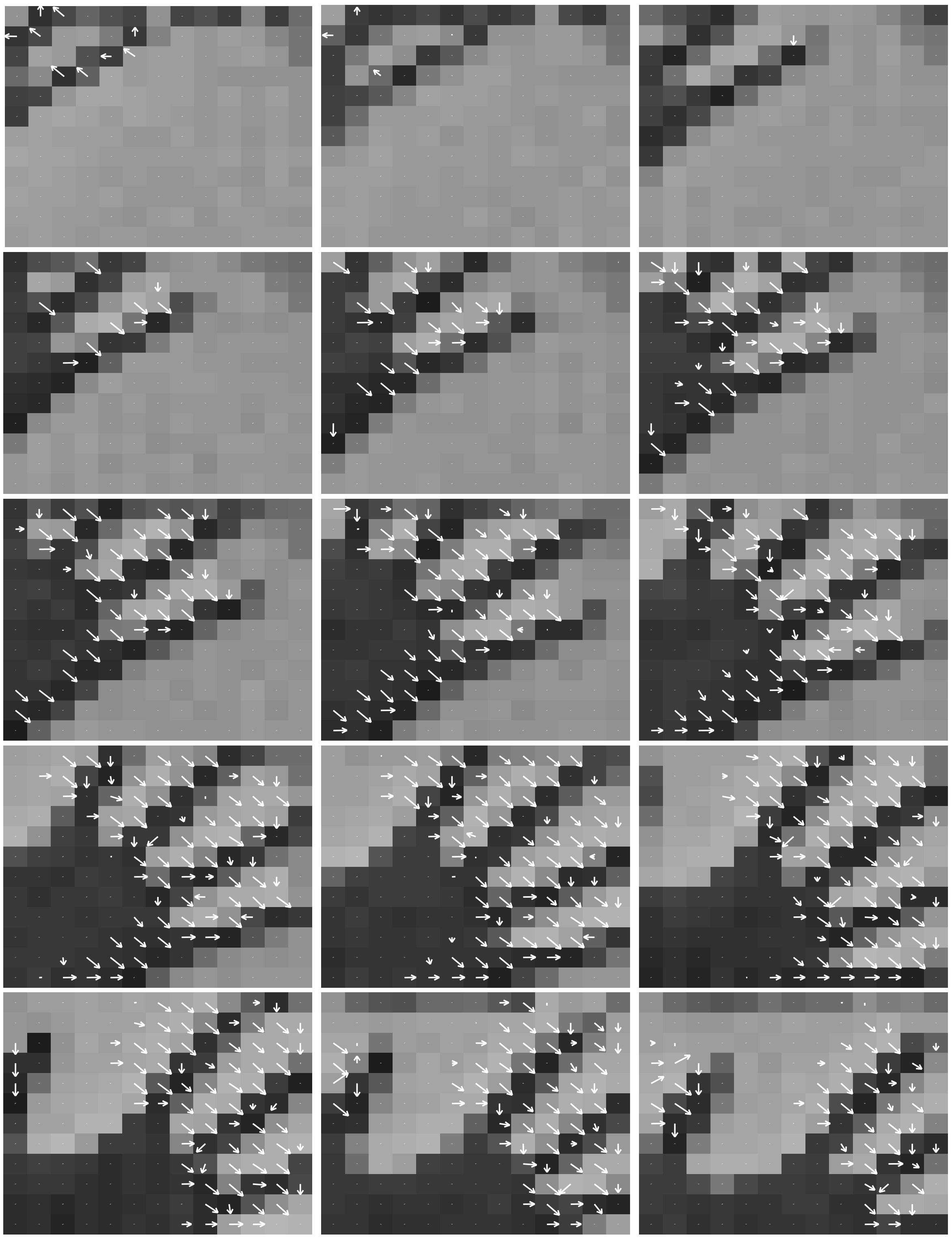


Figure 5.21: FTC flow field: Moving hand.

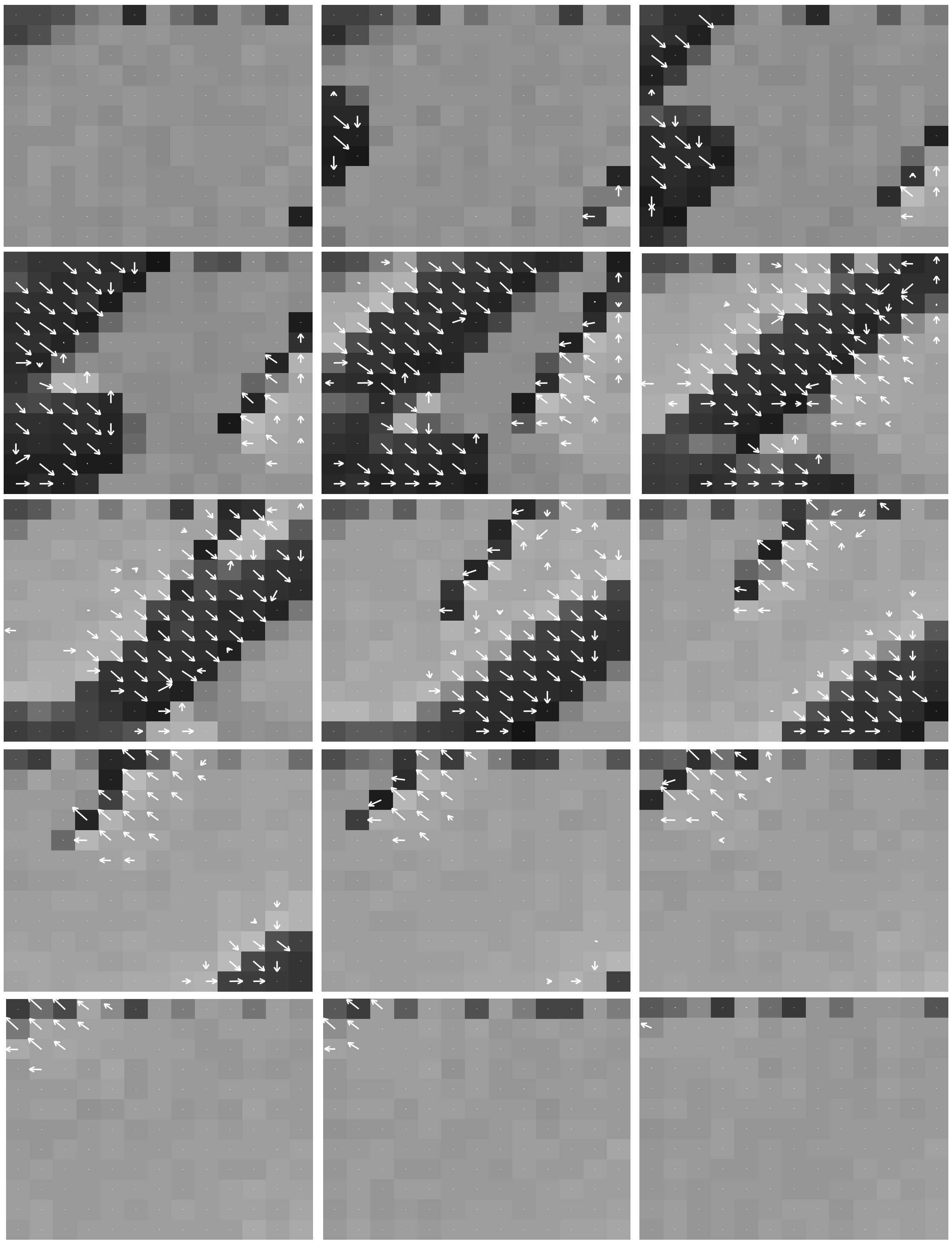


Figure 5.22: FTC flow field: Finger and pen.

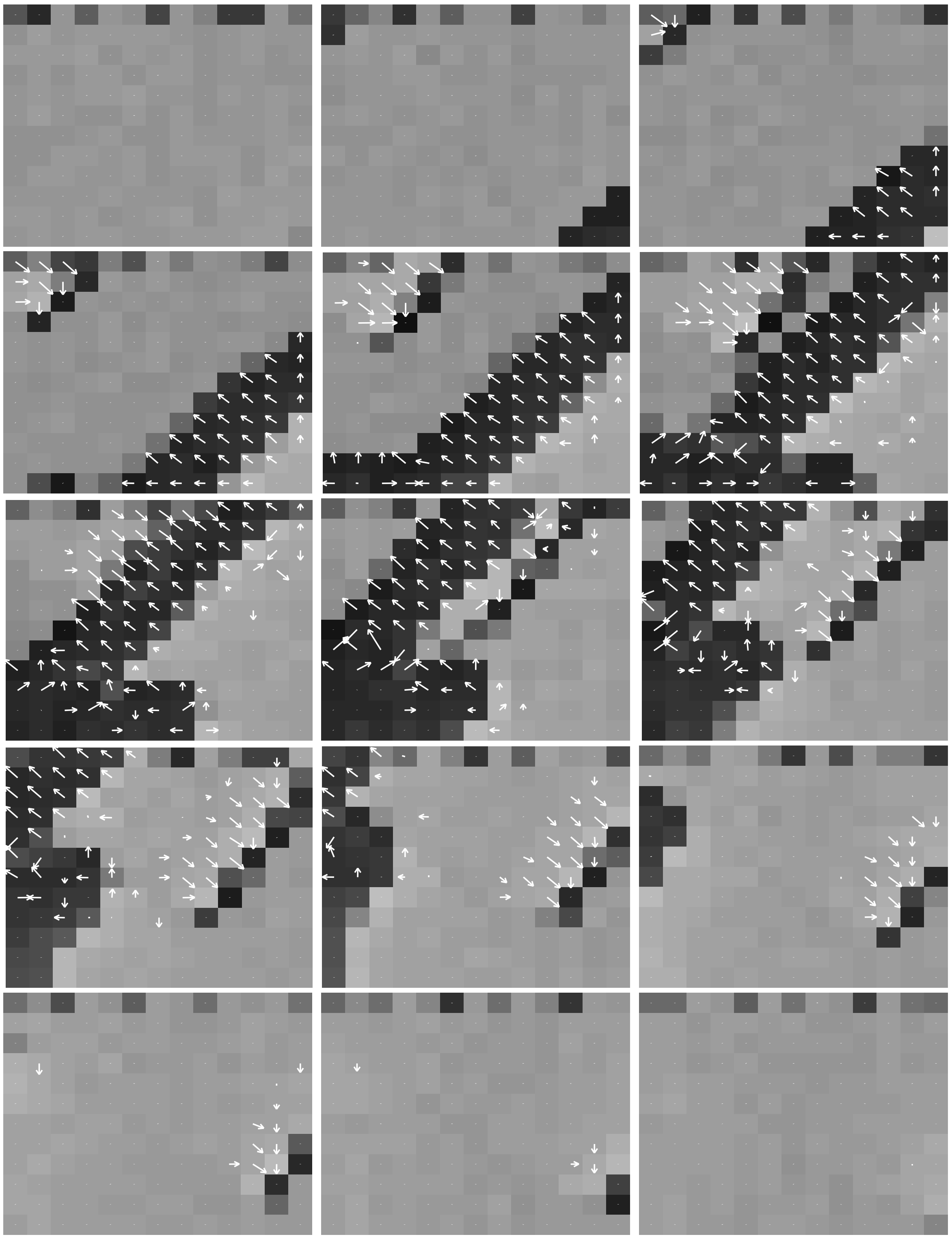


Figure 5.23: FTC flow field: Finger and pen.

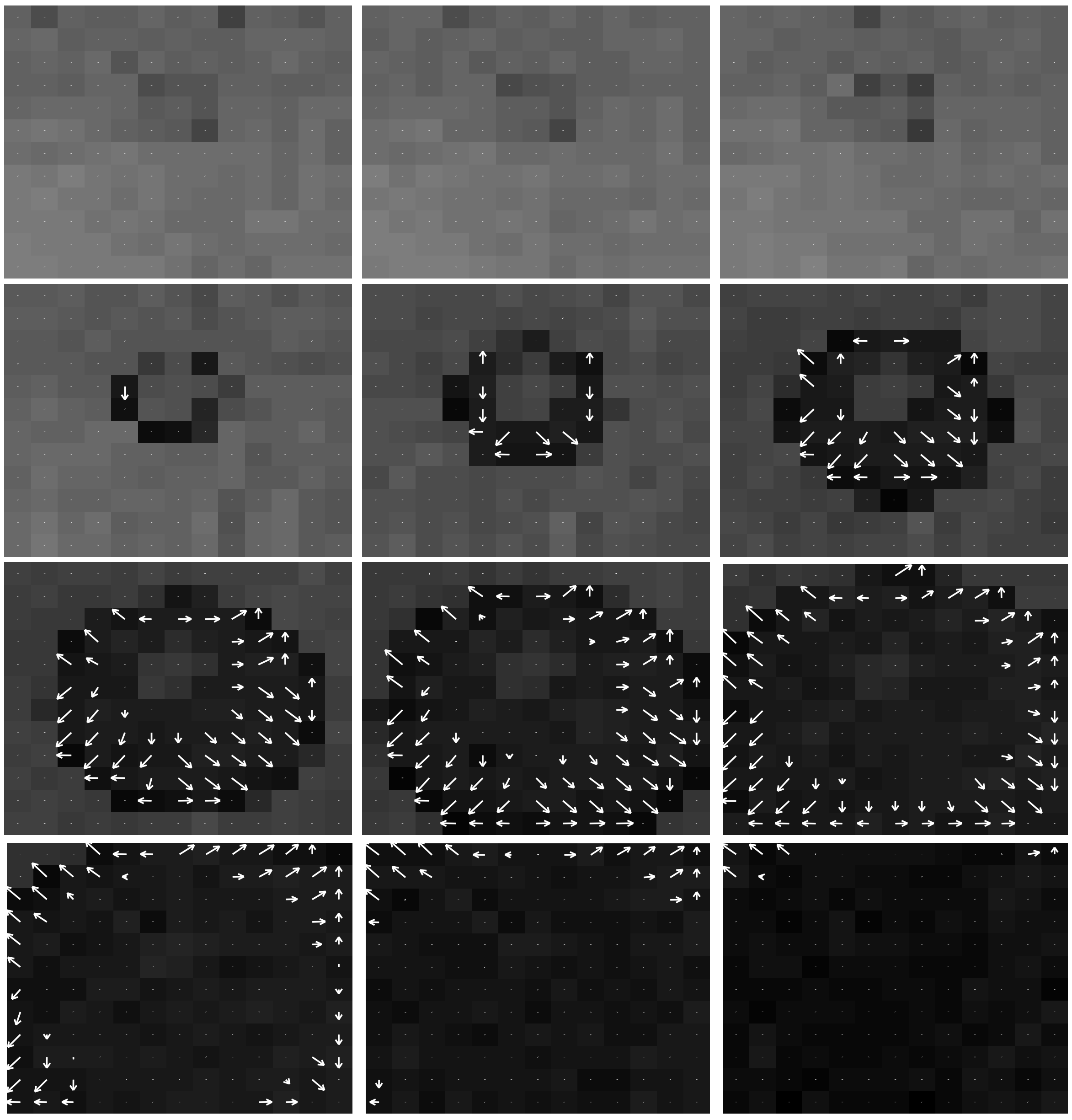


Figure 5.24: FTC flow field: Approaching sphere.

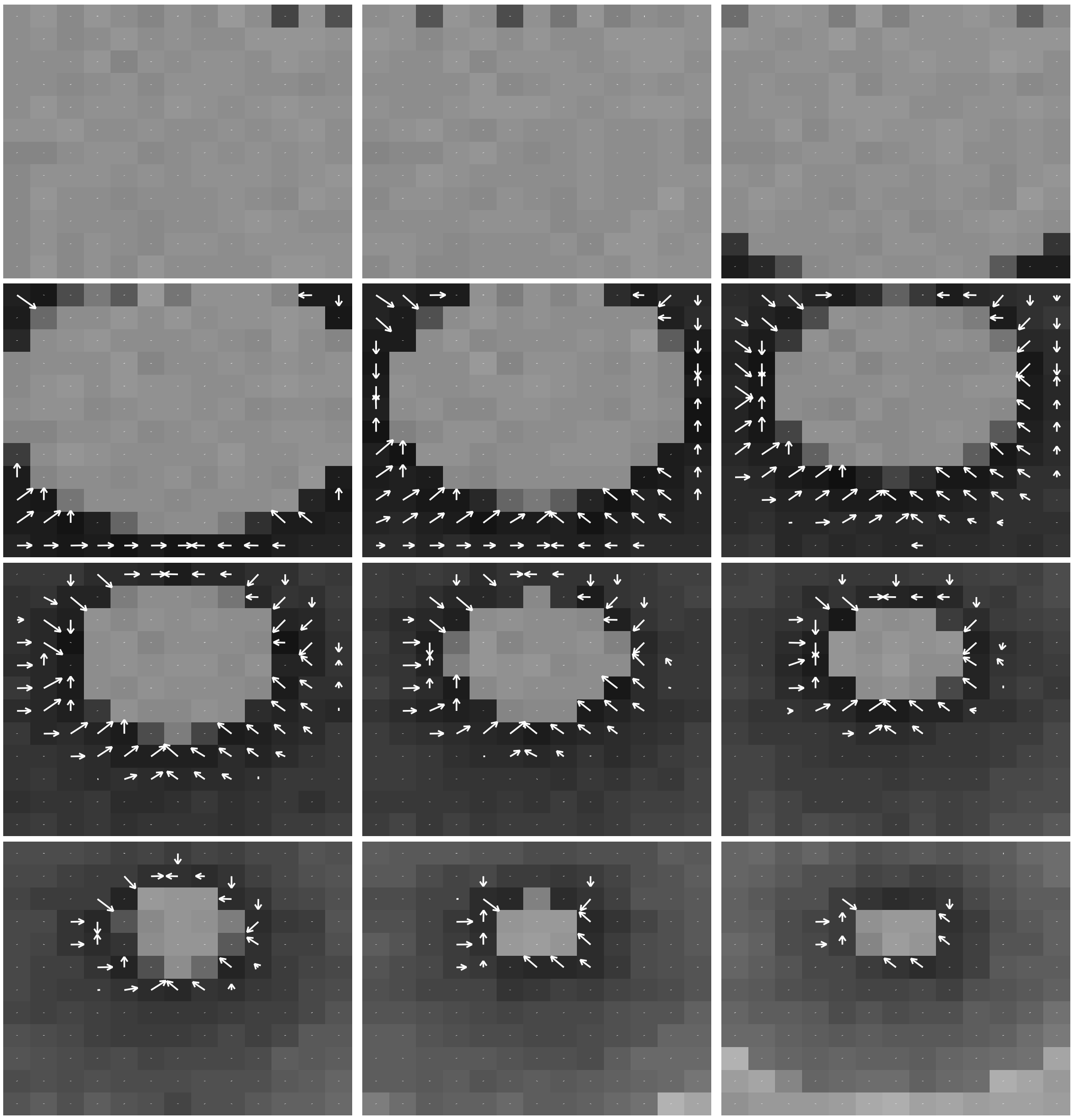


Figure 5.25: FTC flow field: Receding sphere.

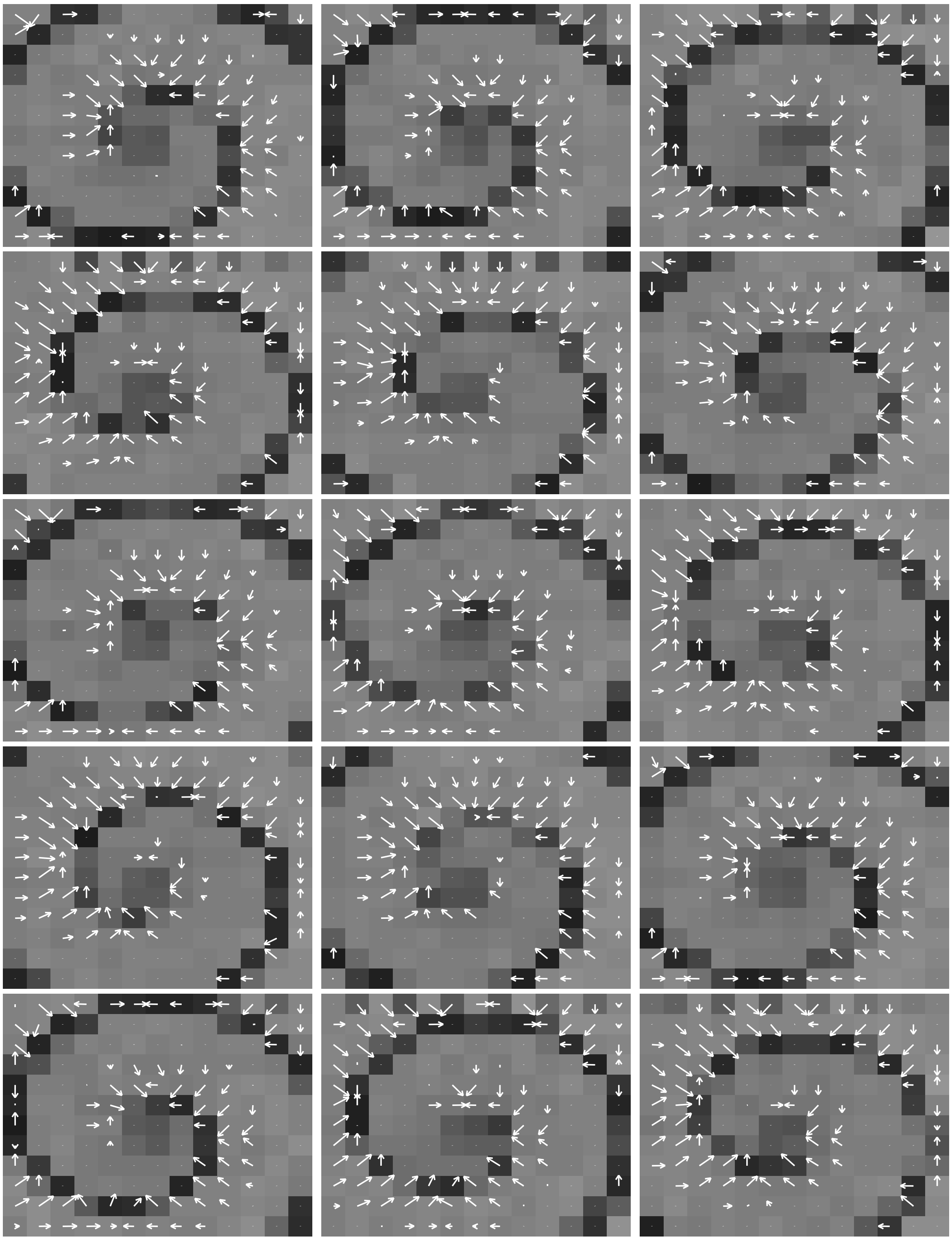


Figure 5.26: FTC flow field: Rotating spiral.

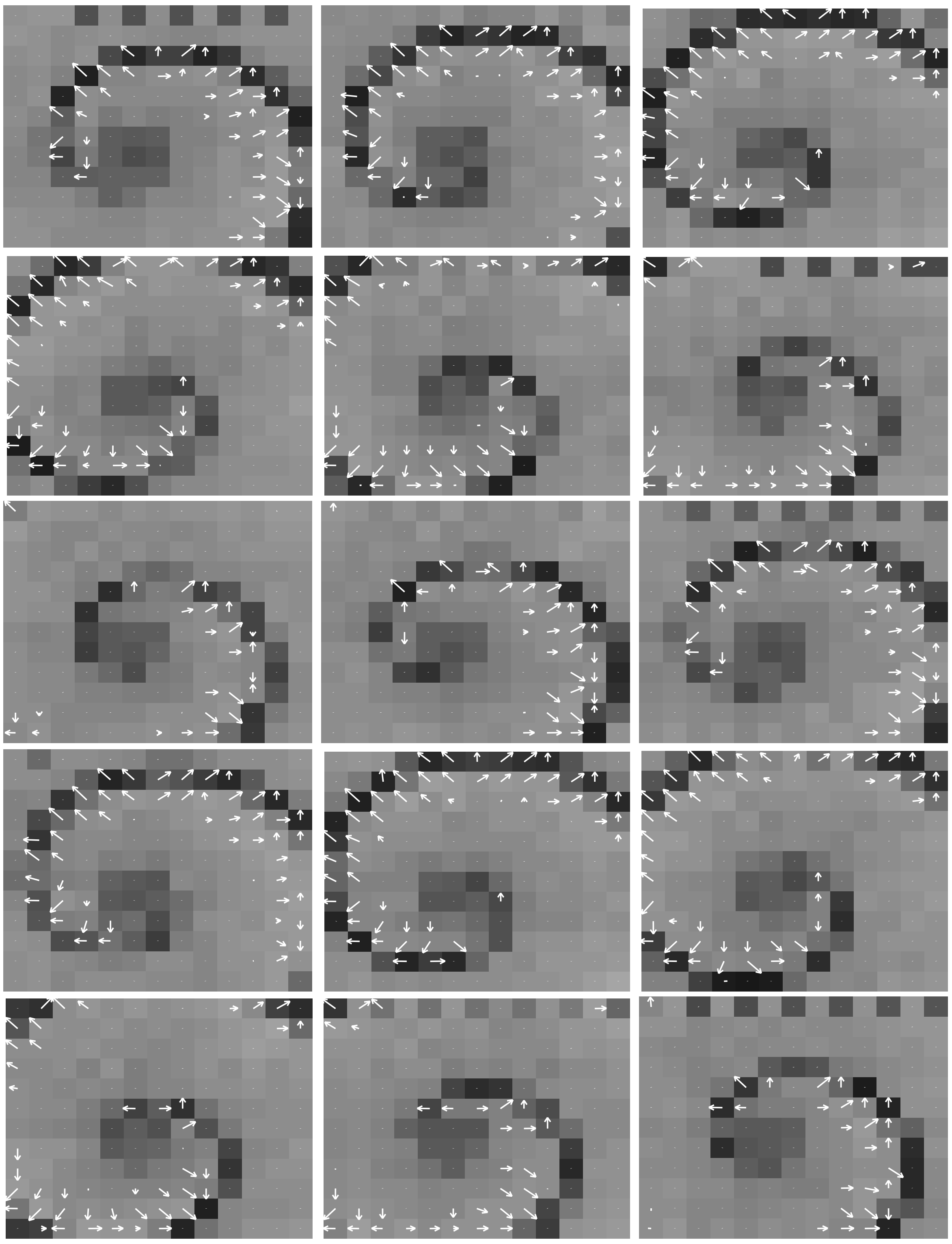


Figure 5.27: FTC flow field: Rotating spiral.

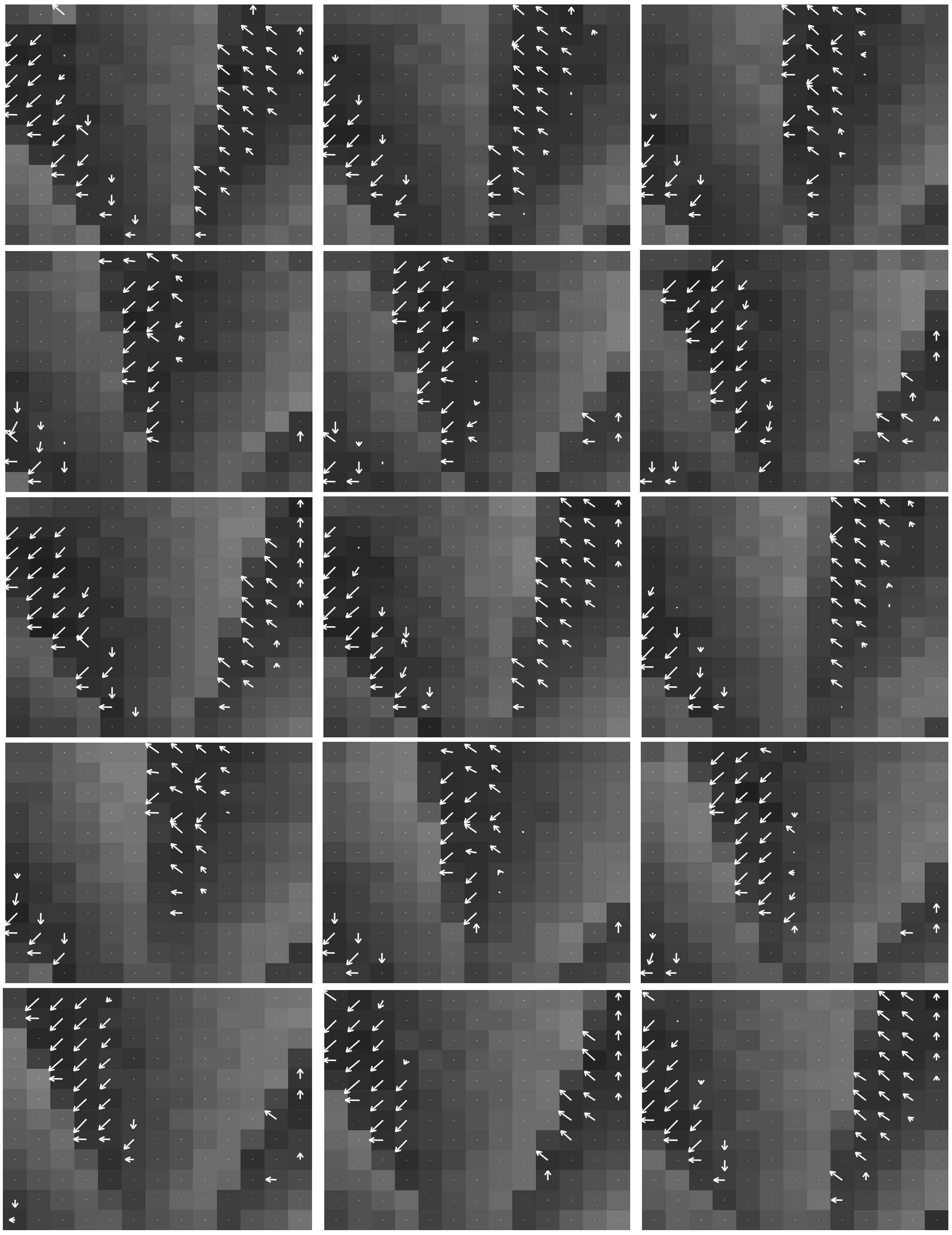


Figure 5.28: FTC flow field: Rotating wheel.

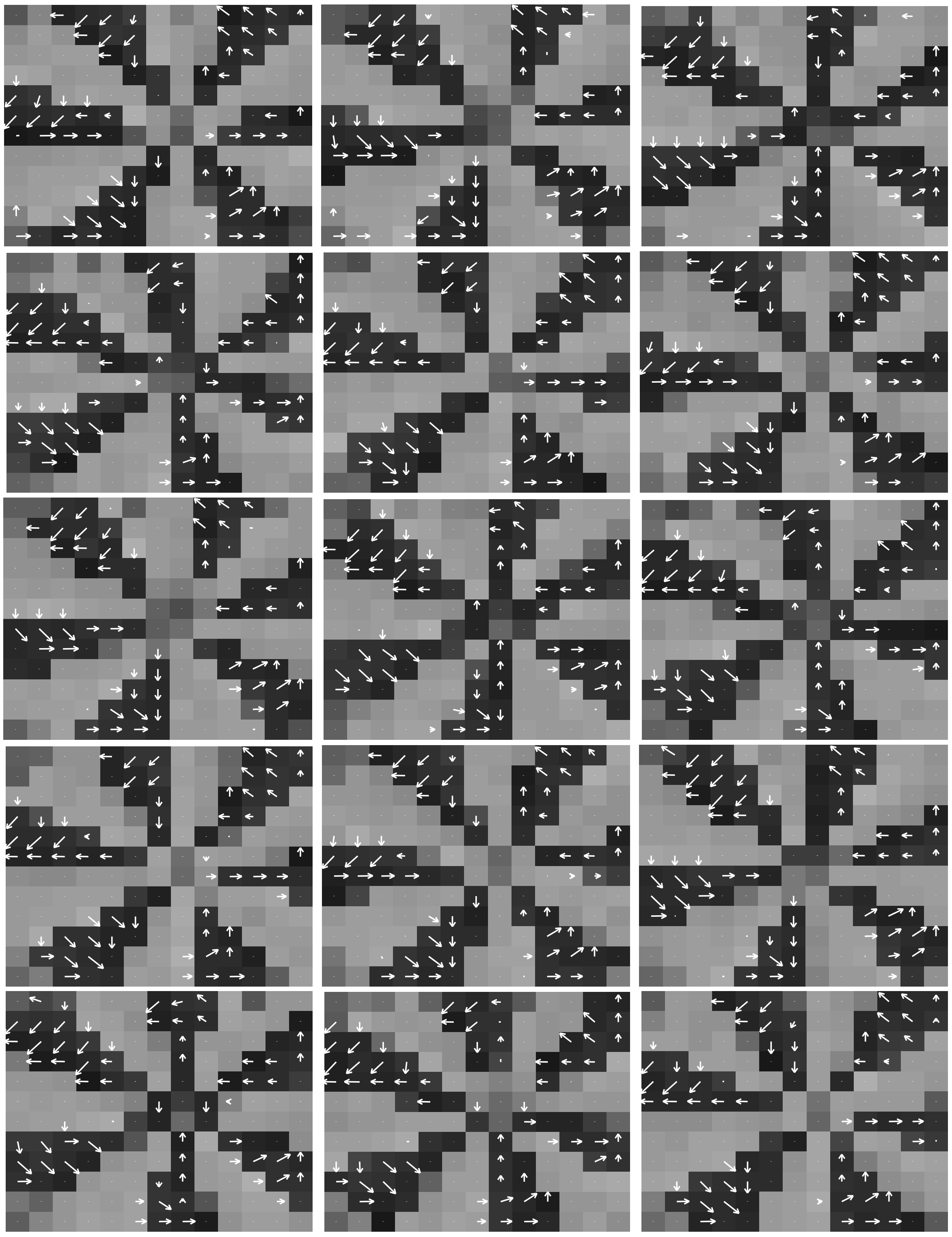


Figure 5.29: FTC flow field: Rotating wheel.

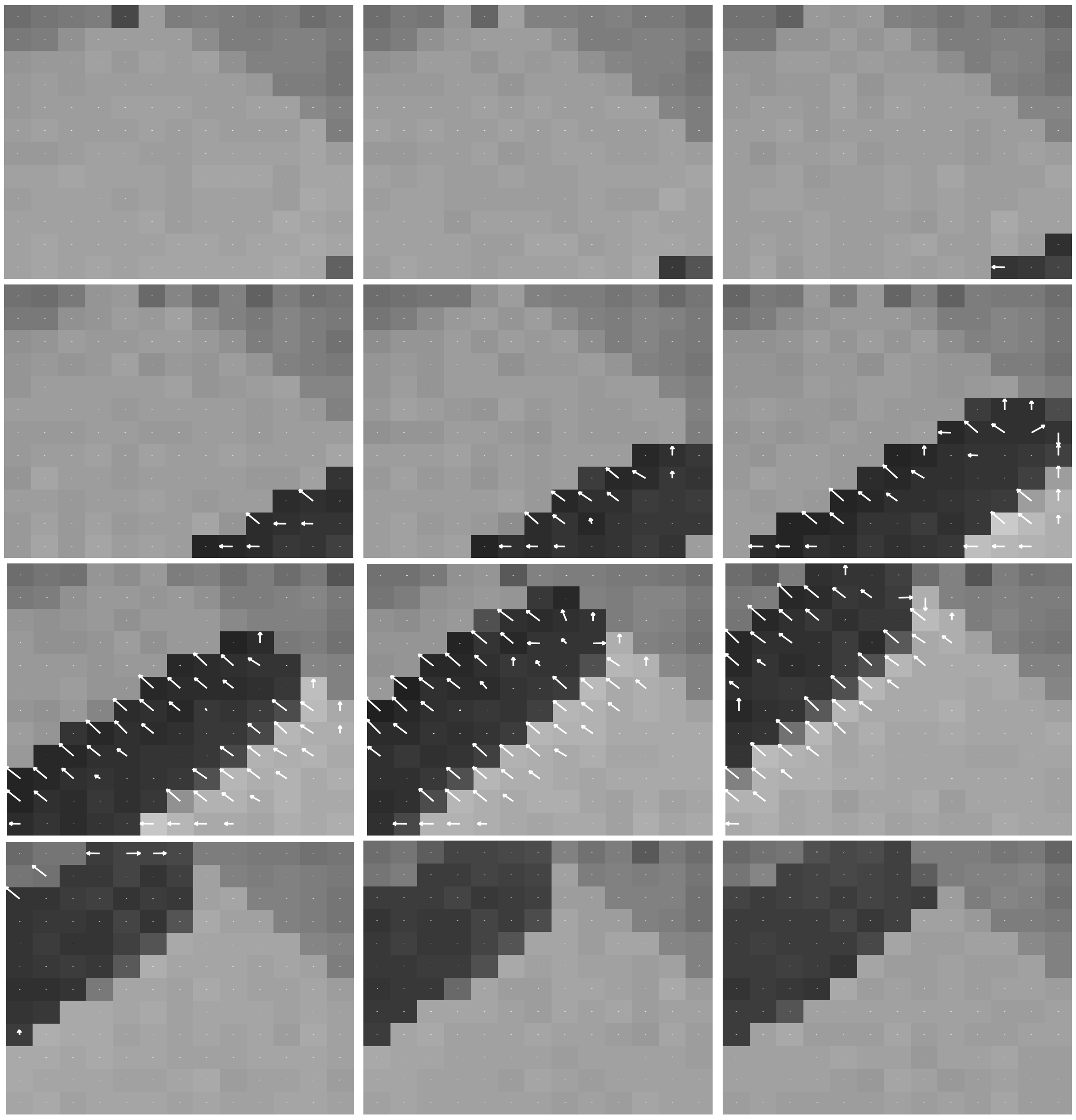


Figure 5.30: FTC flow field: Detecting both edges.

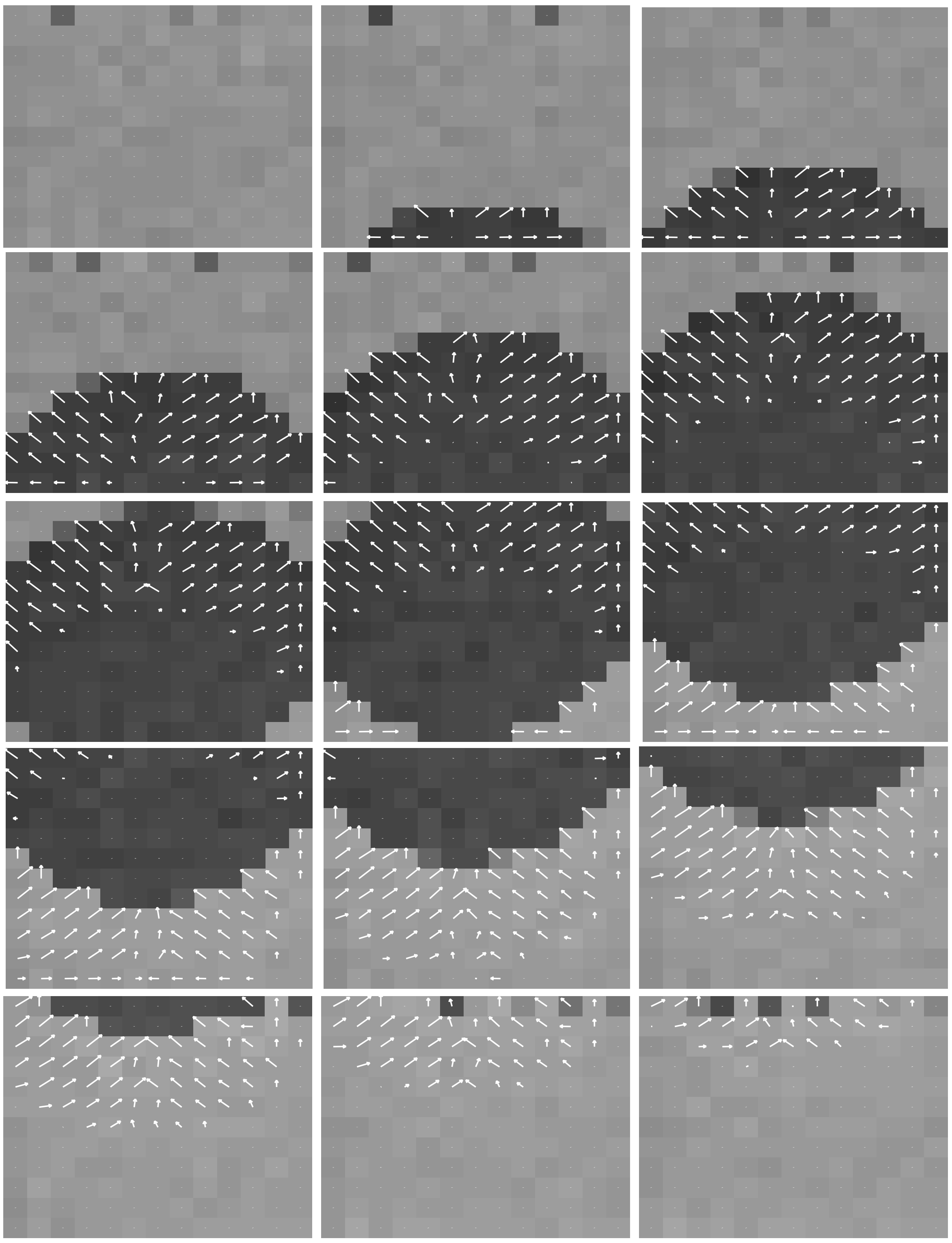


Figure 5.31: FTC flow field: Round object.

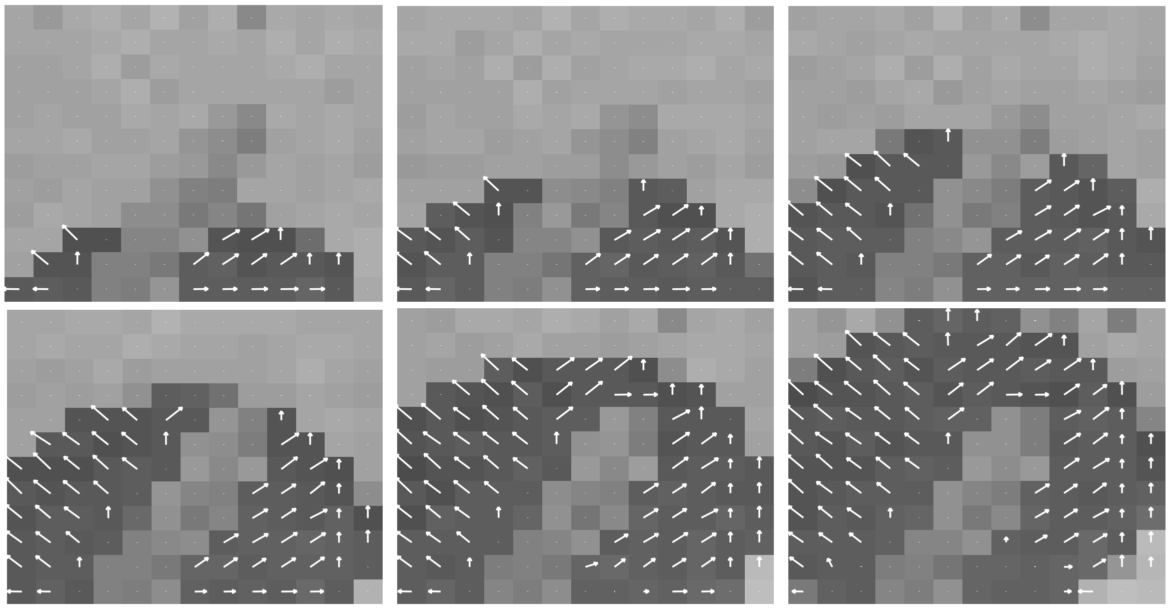


Figure 5.32: FTC flow field: Occlusion.

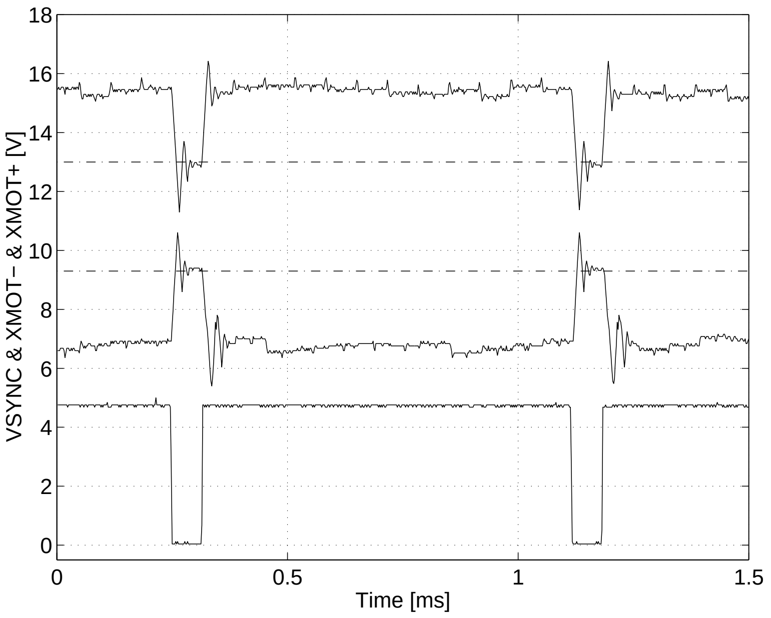
5.8 Remark on Bias Settings

For the operation of the FTC sensor a proper understanding of the bias voltages is crucial. Therefore the most important biases are shortly summarized and commented. The actual bias voltages are dependent on the particular layout.

1. $V_{pthres} = 3.75\,V$ is one of the most important biases because it sets the threshold for TED spike generation.
2. $V_{hysbias} = 4.02\,V$ is a sensitive bias for the temporal edge detector.
3. $V_{prbias} = 4.18\,V$ is the bias for the photoreceptor.
4. $V_{psource}$ is set to Vdd if only bright-to-dark edges are to be detected, should be a few mV lower for detection of both edges.
5. $V_{nsource} = 366\,mV$ sets the amplification of the TED current. Changing $V_{nsource}$ is almost equivalent to changing the threshold V_{pthres}.
6. $V_{pleak} = 3.95\,V$ sets the duration of the facilitation time window.
7. $V_{nleak} = 469\,mV$ sets the persistence time of the output τ_{pt}.

5.9 Remark on the Scanning Speed

The on chip scanners can be driven with a computer through a digital I/O card, or can be driven by a function generator, if the SYNC pulses are inverted and fed back into the BITIN input. It turned out that driving the scanners with a function generator was favorable, because in that case the clock never stopped and less digital noise seemed to be generated. For normal operation the sensor current output was amplified to a voltage signal, converted into digital and displayed or stored through a computer. In this demo mode the fastest frame rates that could be achieved were about 30 Hz. Limiting factors are the speed of the computer and the speed of the AD card. In order to assess the speed with which the sensor output could be read off if faster equipment were given, the sensor output was displayed on an oscilloscope such that the VSYNC signal was used as trigger. That way one column of the chip output could be displayed at a time. In Figure 5.33 the results are shown. For a clock rate of 14.5 kHz (frame rate 80 Hz) the motion output indicates unambiguously the direction-of-motion for all pixels. For a clock rate of 56 kHz (frame rate 307 Hz) transients occur after abrupt signal changes. This is most visible after the SYNC period, where the motion output goes to the null level, indicated by the dashed line. The reason why there is a lag after fast signal changes is most likely not in the sensor, but caused by the limited speed of the operational amplifiers. For the FTC sensor board cheap LM324 were used and already driven to the limits by amplifying currents as low as nano Amperes. Higher frame rates than 300 Hz should be achievable with faster current amplifiers, such as the TL074. The important finding of this high speed experiment is, that the sensor performance does not degrade for higher scanning speeds, thus the sensor can bear much higher frame rates than what could be handled by the present external equipment.



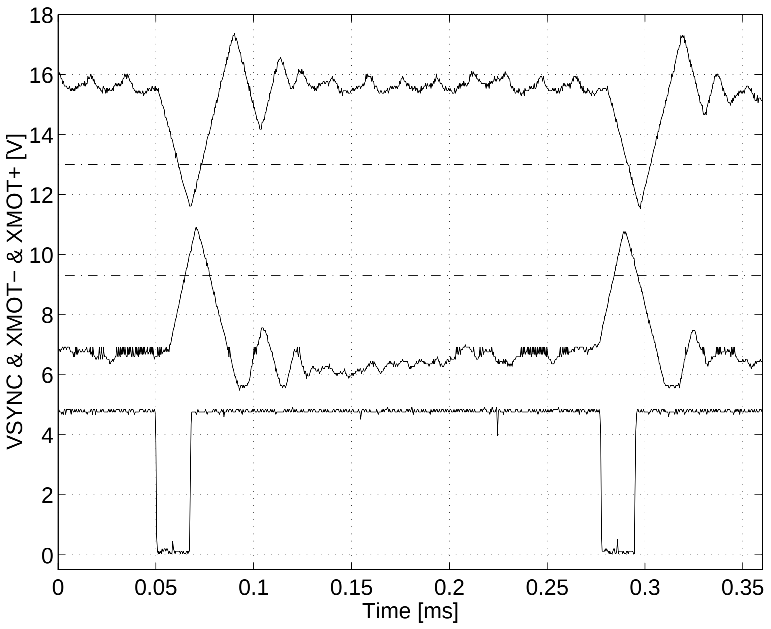


Figure 5.33: Demonstration of the influence of the clock rate on the motion output. The top traces of both figures show the motion output of a column of 12 pixels for motion in positive X direction. The second traces show the motion output for the opposite direction-of-motion. During the SYNC signal (bottom trace) the motion output is zero, indicated by the dashed baseline. On the top figure the clock frequency was 14.5 kHz, corresponding the a frame rate of 80 Hz, for the bottom figure the clock rate was as high as 56 kHz, corresponding to a frame rate of 307 Hz. For high scan rates the signal transients become dominant.

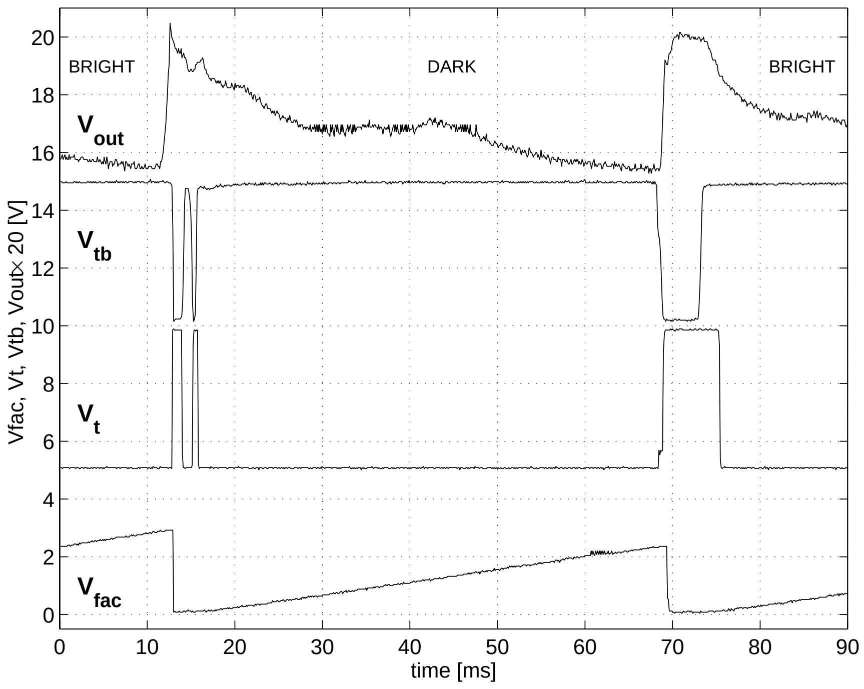


Figure 5.34: Limitations of the FTC sensor. A black thin bar was moved fast over the sensor. The top trace shows the non thresholded TED output V_{out}. The trace is shifted and multiplied by 20 for clarity. The second trace shows the thresholded pulse V_{tb}, shifted up by 10 V, the third trace the inverse V_t, shifted up by 5 V and the bottom trace shows the facilitation voltage V_{fac}. Several limitations can be explained from this plot, see text.

5.10 Sensor Limitations

In this section the limitations of the FTC sensor will be discussed. The limitations are of two kinds. First the sensor operation is limited by the circuit performance, most importantly the temporal edge detector. Second, even if the algorithm were implemented perfectly and the circuits were behaving ideal, due to the very FTC algorithm theoretical limits on the speed and the spatial frequency of stimuli can be derived and effects on the 2-D performance can be deduced.

5.10.1 Circuit limitations

The the following the circuit issues that have been found to limit the sensor operation are listed and possible solutions are discussed:

1. **Different sensitivity to both edges.** In the FTC sensor the analog TED output V_{out} is thresholded for both edges with the same threshold V_{pthres}. Unfortunately V_{out} not only changes its peak height for one edge depending on the stimulus velocity, but the change of V_{out} for the other edge is different. As a consequence the threshold V_{pthres} can be set for one velocity such that both edges are detected with equal strength, but for a different velocity one edge

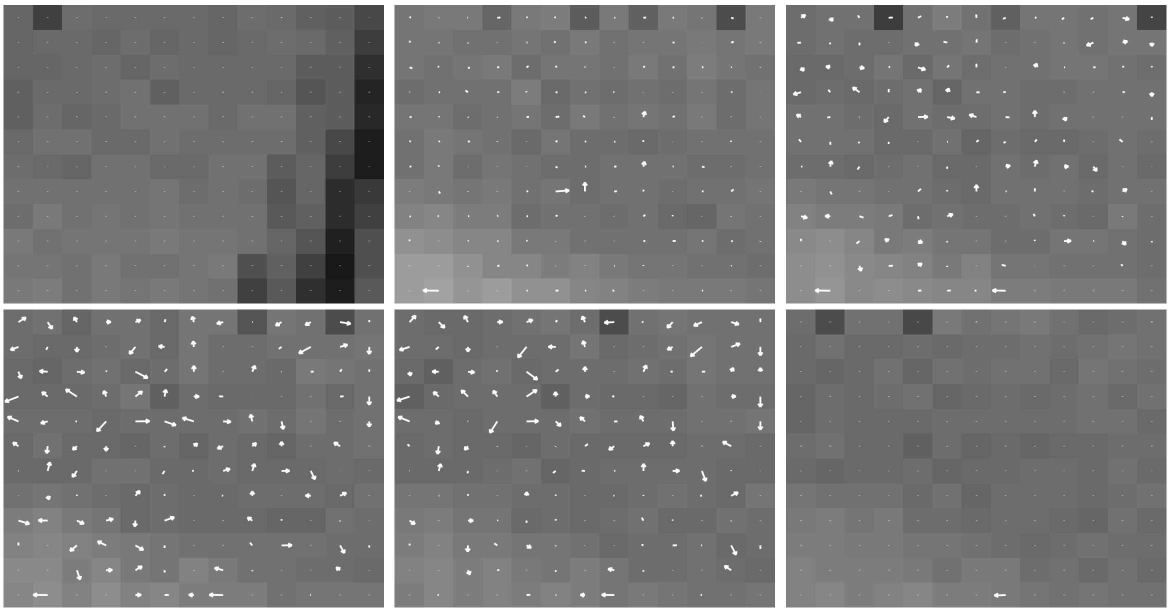


Figure 5.35: 'Blooming' of the vector field due to device mismatch. A hand was passed over the sensor very fast, in the first snapshot the hand enters from the lower right corner. The frames are 110 ms apart. In the subsequent frames vectors seem to grow out of nowhere pointing in random directions. The reason is that during stimulus presentation the TED spikes overlapped and all direction-of-motion indicators were triggered, which from the comparator does not yield zero output because the leakage times τ_{fac} of the pixels are not identical due to device mismatch.

might cause a stronger peak in V_{out} and the other edge a weaker one. In this case only the first edge would be detected after thresholding, whereas the latter would be lost. Although in Figure 5.34 both edges are detected, it can be seen that the peak height of V_{out} is different for both edges. With a higher threshold only one edge would have been detected.

2. **Spike length.** The length of the TED spike determines the fastest stimulus velocity to which the FTC sensor can respond. If spikes from neighboring pixels overlap then the two corresponding direction-of-motion indicators are activated at the same time and the output of the comparator should be zero. Due to device mismatch, though, the leakage times τ_{pt} can be slightly different and the comparator output in this case would increase in magnitude. This is visible as 'blooming' of the vector field during the time τ_{pt} after a fast stimulus presentation (see Figure 5.35). As can be seen in Figure 5.34 the spikes of both edges can have considerably different lengths. In this case one edge would cause incorrect outputs whereas the other edge would still be causing correct outputs. The TED spike length for a given edge is determined by the threshold. For weak thresholds the spikes are longer.

3. **Double spikes.** For TED threshold adjusted for very low contrast edges instead of only one spike for an edge, two or more spikes can be generated for a fast and high contrast edge. This is the case for the falling edge in Figure 5.34. If the inter spike time of the double spike is comparable to the inter-pixel travel time of the stimulus then wrong direction-of-motion outputs can occur. This effect, again, is determined by the sensitivity of the TED.

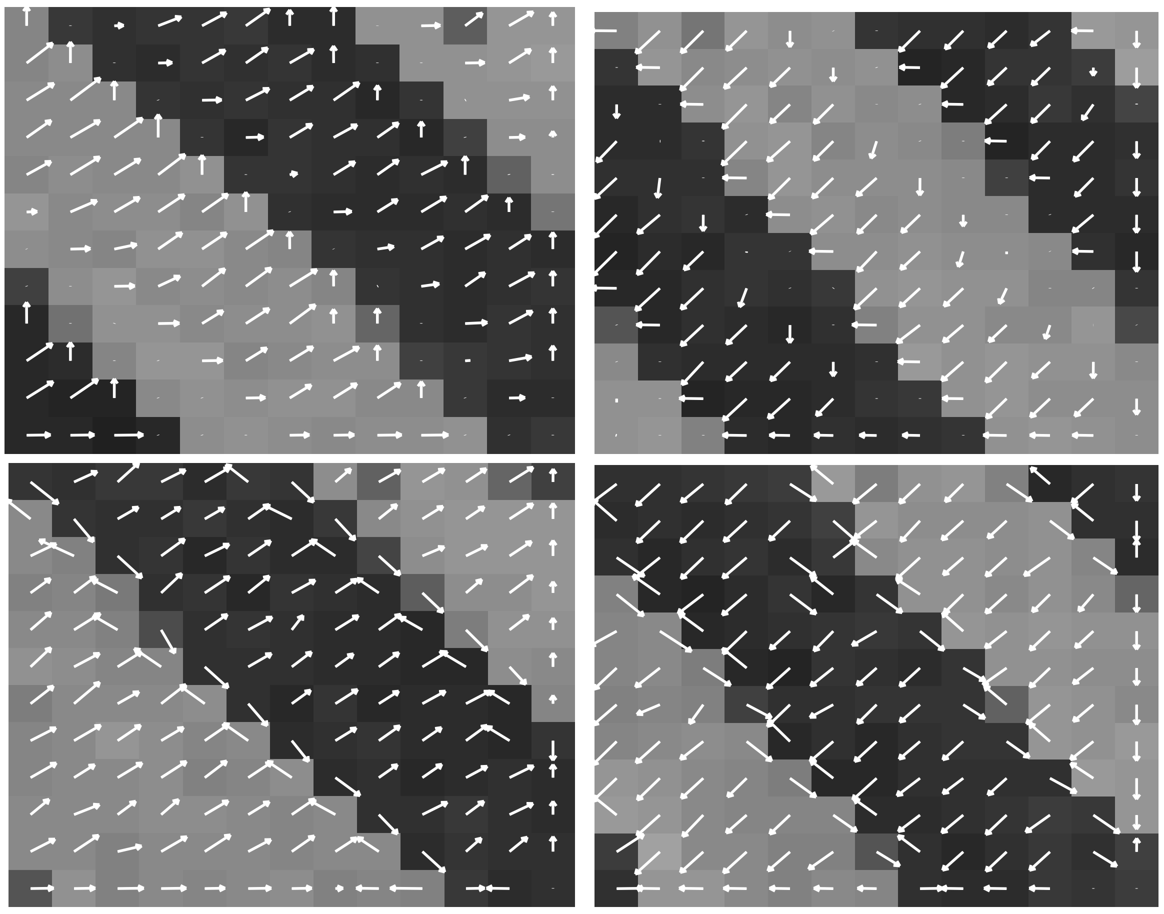


Figure 5.36: Example of pseudo aliasing. The top snapshots show the correct output for a bar pattern, both edges are detected. The bottom snapshots show a bar pattern of higher spatial frequency or, equivalently, the same bar pattern detected with longer system time constants τ_{fac} and τ_{pt}. The flow field is correct except at the edge, where either X or Y component or both are reversed for one inter pixel travel time. As explained in the text and visible in the figure, after this time the correct output is generated.

All the above mentioned effects are caused by the fixed sensitivity of the TED and the common threshold for both edges. Although the analog VLSI designer tries to minimize the amount of required external bias voltages, in this case the introduction of an additional threshold voltage could be favorable. If both edges were thresholded with separate biases, the TED sensitivity could be adjusted separately. But the only way to cope with the fact that the height of the TED analog output V_{out} changes with temporal contrast differently for both edges could be adaptable thresholds. The idea would be that the circuit kept its mean firing rate at a constant level. For low contrast visual scenes the threshold would become weaker, for high contrast scenes the threshold would become stronger.

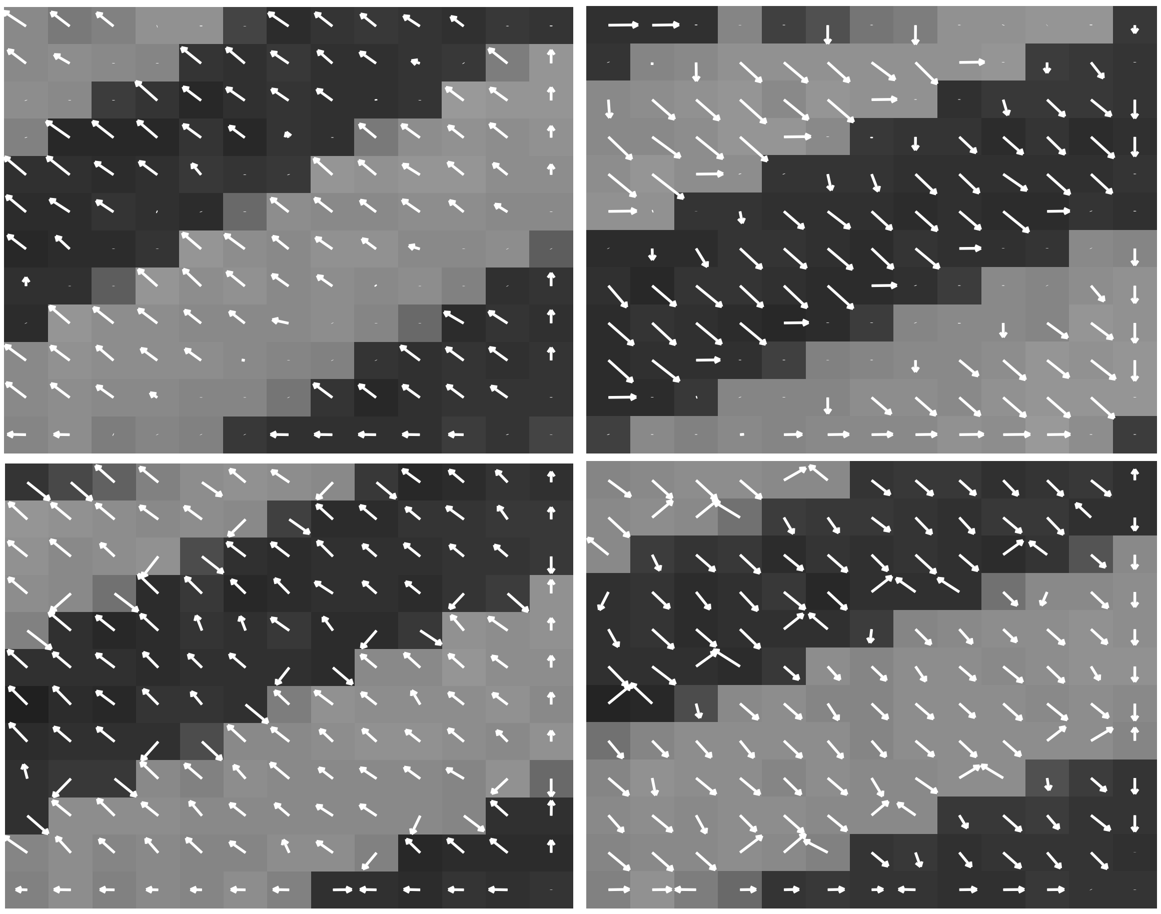


Figure 5.37: Example of pseudo aliasing. The top snapshots show the correct output for a bar pattern. The bottom snapshots show the effect of pseudo-aliasing. At the leading and trailing edge of the bars null direction vectors are generated. Between the edges the correct output is produced.

5.10.2 1-D effects of the FTC algorithm

In the following the limitations on speed, spatial frequency and size of stimuli are derived, which are caused by the FTC algorithm which is effectively a 1-D algorithm but also affects the 2-D performance of the sensor. For the derivation several time constants have to be distinguished: On the shortest time scale is the TED spike length τ_{spike}. The facilitation time τ_{fac} and the persistence time τ_{pt} have already been introduced in section 5.3. The time it takes an edge to travel from one pixel to the next (inter pixel travel time) is called τ_{ip}. The inter pixel travel time is inversely related to the stimulus velocity. τ_{edge} is defined as the time between spikes caused by either the same edge of two objects (if the TED is sensitive only to one edge) or both edges of one object (if the TED is sensitive to both edges). In the latter case τ_{edge} is dependent on the stimulus width.

If the stimulus moves so fast that it reaches the next pixel within the time of the last TED spike, then both direction-of-motion indicators are triggered and the motion output is zero. If the edge travels so slow that it reaches the next pixel after the facilitation time is over, also no motion is

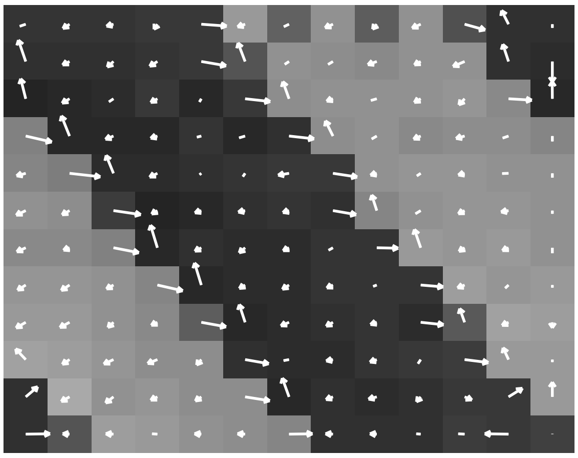


Figure 5.38: Pseudo aliasing and long persistence time. The spatial frequency and speed of the moving bar pattern are too high for the system time constants τ_{fac} and τ_{pt}. In effect null direction output is generated at the edges, as in the previous figure, but the output is zero otherwise.

reported. The condition on the edge inter pixel travel time for a motion output is therefore:

$$\tau_{spike} < \tau_{ip} < \tau_{fac}. \tag{5.1}$$

With typical times $\tau_{spike} = 1\,ms$ and $\tau_{fac} = 100\,ms$ a theoretical dynamic range for the FTC sensor of two orders of magnitude of velocity can be achieved. As can be seen the FTC algorithm was designed such that beyond these limits the motion output is kept zero, rather than reporting the null direction output.

The following analysis is concerned with the interaction of spikes generated by different edges. If the inter edge time τ_{edge} is smaller than the facilitation time τ_{fac} then for the second edge the facilitation caused by the first edge has not yet decayed. As a result for the time τ_{ip} the second edge can cause a null direction output; this effect is called here ‘pseudo aliasing’ and is shown in Figure 5.36. Therefore the FTC algorithm requires the additional constraint

$$\tau_{edge} > \tau_{fac}. \tag{5.2}$$

For $\tau_{fac} = 100\,ms$ this means that on one pixel up to 10 edges per second can be presented without causing wrong outputs. For faster edge frequencies even if null direction outputs have been reported during the time τ_{ip}, after that time the output will be either zero or correct, as is shown now. In circuit simulations it is found, that the inputs to the comparators only have to differ by 10 mV in order to generate a direction-of-motion output current significantly greater than zero. As explained earlier (see section 5.3) the direction of motion outputs are triggered to Vdd=5 V and decay linearly over the time τ_{pt}. Therefore if the trigger times of the direction-of-motion indicators differ by more than $\tau_{pt} \times 10\,mV/5\,V$ the correct direction-of-motion is reported:

$$\tau_{ip} > \tau_{pt}\,\frac{10\,mV}{5\,V} \tag{5.3}$$

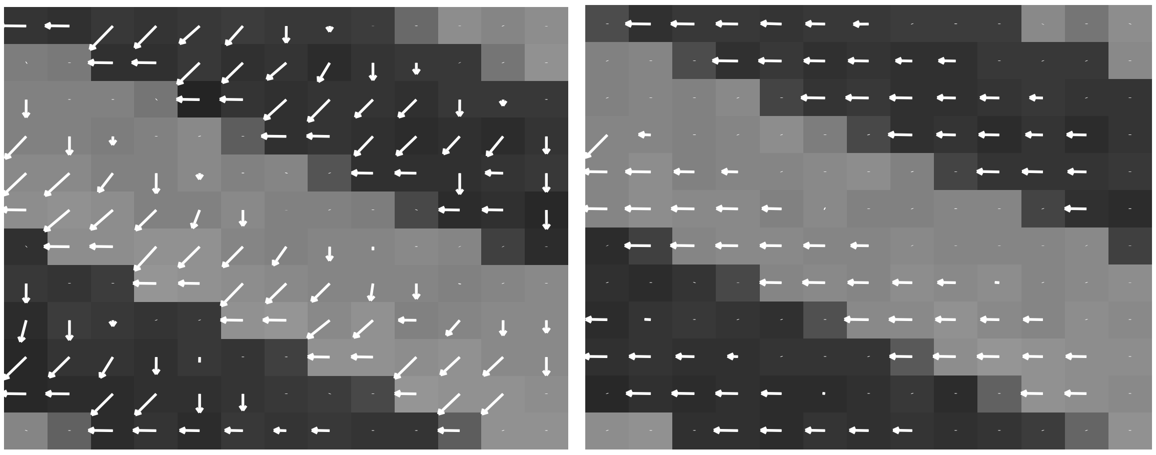


Figure 5.39: 1-D limits in the 2-D flow field. A bar pattern was moved down left. The sensor was sensitive to both edges. On the left snapshot on-transients and off-transients are visible, on the right the edge velocity in Y direction was too slow, only X components of the direction-of-motion vectors are generated.

For a persistence time such that even for the slowest stimulus velocities the direction-of-motion output is kept by five pixels in a row, $\tau_{ip} = 5\,\tau_{fac}$, this condition reads

$$\tau_{ip} > 5 \times 100\,ms\,\frac{10\,mV}{5V} = 1\,ms \tag{5.4}$$

which interestingly is the same figure as for the earlier condition $\tau_{ip} > \tau_{spike}$ and therefore does not represent any further restriction. In Figure 5.38 the case is shown where the persistence time τ_{pt} was made so long that condition 5.3 was violated. This can be the case if the flow field is to be kept on the sensor array for a long time. To avoid this scenario condition 5.2 should really be obeyed.

5.10.3 2-D effects of the FTC algorithm

In this paragraph effects of the limitations explained above on the 2-D performance of the FTC sensor are discussed. The effects are twofold. First there are transient motion vectors being generated at the very front of every edge. This in fact is intrinsic to every discrete feature based 1-D algorithm that is extended to 2-D by just duplication of the 1-D circuitry. Second, specific to the FTC algorithm, the flow field can be distorted for slow velocities.

Transient motion vectors in 2-D

For any feature based algorithm that computes the X component and the Y component of the motion vector separately this motion vector will be transient at the very edge of a moving feature. First the motion vector will be zero, then one of the components is being reported, the other component still being zero. This phase is called transient in this context. Only after the feature moved far enough that also the second component of the motion vector could be determined, the correct motion vector is reported. The FTC sensor like any other reported aVLSI feature based sensor consists of two 1-D sensors. These sensors report the 1-D direction-of-motion independently and cause the transient 2-D output.

The FTC sensor, specifically, is not point symmetric in a 2-D context. This means that for an edge moving up right, up left, down right or down left the transients look differently. For the up right movement *always* the positive Y component is determined first, then the positive X component. For an edge moving up left there are is *no* transient motion output, the vector points in 135^o as soon as the edge passes the pixel. For down right movement the positive X component is determined earlier for edge orientation from 270^o to 315^o, the negative Y component is determined earlier for edge orientation from 315^o to 360^o. Finally for down left movement *always* the negative X component is determined first. All these cases can be observed in the sample outputs shown throughout this chapter.

In contrast to these 'onset' transient vectors, due to the finite persistence time there are also 'offset' transient vectors. For the FTC sensor every direction-of-motion indicator is held for the time τ_{pt}, after which the corresponding direction-of-motion vector component is decaying to zero. As a result, the 'offset' transients have the exact opposite characteristic to the onset transients described above. This can be observed for example in Figure 5.39 left, where an edge is moving down left. The 'onset' vector transients point in negative X direction, whereas the 'offset'transients point in negative Y direction.

'Onset' transient vectors are generated only during the time of travel of the edge between two neighboring pixels, which can be from milliseconds to hundreds of milliseconds. 'Offset' transients are visible during the decay time of tens of milliseconds. Thus the transient times are usually very short compared to the stationary direction-of-motion output. It will depend on the application that uses the flow fields if transient vectors are problematic. By visual inspection of the flow fields for scanned outputs the transients are barely perceived.

Normal vs. orthogonal motion vectors

In section 5.10.2 the condition for obtaining correct direction-of-motion output $\tau_{spike} < \tau_{ip} < \tau_{fac}$ was derived for the 1-D case. For the 2-D sensor this extends straightforwardly to

$$\begin{aligned} \tau_{spike} < \quad & \tau_{ip}^{x} \quad < \tau_{fac} \\ \tau_{spike} < \quad & \tau_{ip}^{y} \quad < \tau_{fac}. \end{aligned} \tag{5.5}$$

Different cases can be distinguished:

- *Both inequalities hold.* In this case, except for the transient period, diagonal vectors are generated.
- *Motion too slow in one dimension.* For a square grid implementation of a sensor that relies on time-of-travel, for any given normal velocity of an edge a direction-of-motion angle can be found such that in one dimension the time-of-travel becomes arbitrary small. For the FTC sensor this means that for example for an edge moving down left close to 270^o even though the inter pixel travel time for the Y direction is too long $\tau_{ip}^{x} > \tau_{fac}$ the travel time in X direction

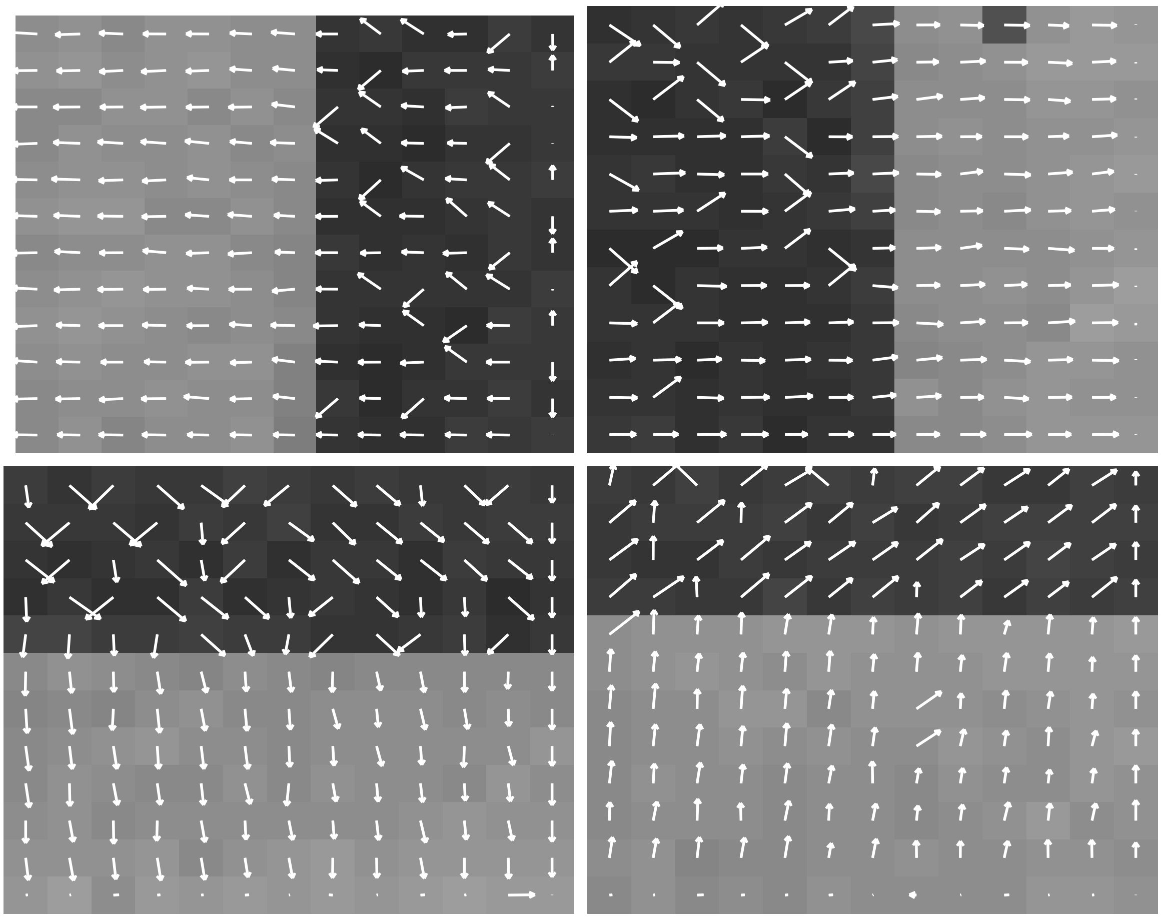


Figure 5.40: Movement in orthogonal grid directions. A bar pattern was moved in the four cardinal directions. The sensor was sensitive to both edges. It can be seen that motion vectors in stimulus direction are generated as well as with vectors in both adjacent diagonal directions.

satisfies the condition. As a result vectors pointing in negative X direction are generated for an edge moving almost in the negative Y direction (see Figure 5.39).

- *Motion too fast in one dimension.* For the same reason an edge can obey condition 5.5 in one grid direction but move too fast in the other direction such that the inter pixel travel time is shorter than the TED spike time: $\tau_{ip} < \tau_{spike}$. In this case, as discussed before, the FTC sensor outputs zero in the concerned direction. This is a desired feature, though, because for edge movement in one of the four cardinal directions the direction-of-motion vector is also then pointing in that direction. This mechanism is the only way the FTC sensor can produce vectors pointing in a different direction than diagonal to the square grid. In Figure 5.40 snapshots of the FTC sensor output are shown for an edge moving in the four orthogonal directions. It can be seen that even for movement in an orthogonal direction diagonal vectors are generated. This is because the TED spikes are very short and are not overlapping. The vectors that point in stimulus direction are caused by overlapping TED spikes orthogonal to that direction. Another interesting observation can be made: 'Off' edges seem to cause less orthogonal vectors than ‘on’ edges. The reason for this phenomenon is, that the TED spikes for ‘off’ edges are shorter than the TED spikes for ‘on’ edges and therefore less likely to overlap.

5.11 Summary

In this chapter it has been motivated that velocity sensors necessarily require components that make their pixels large, which makes it difficult to implement high density arrays. Therefore an algorithm for the computation of the direction-of-motion was developed, which allows for a very compact implementation in analog VLSI hardware. The direction-of-motion output is computed as a vector which can assume one out of four directions for an ideal TED. For finite TED spike lengths the output points in one out of eight directions. The algorithm also deals with circuit noise in such a way that rather no motion output is generated than null direction output. No clock is required in the pixel and no reset is necessary to clear signals, which further increases the robustness. The algorithm was implemented in analog VLSI hardware in a standard 1.2μm technology. A fully functional test chip was presented and results from the 13×12 pixel array were shown. The sensor was proven to be sensitive to contrasts as low as 10% and operates reliable for contrasts larger than 40% over a velocity range of well over one order of magnitude, from 25 pixels/sec up to 450 pixels/sec. For high contrasts the sensor was shown to work with 100% reliability down to velocities of 0.5 pixels/sec. The orientation tuning curve matches the theoretical prediction if the finite spike length of the TED is taken into account. Image sequences of the sensor direction-of-motion output and the gray value image were presented, where natural stimuli such as fingers and a hand were used. The flow fields suggest several possible applications of the FTC sensor, amongst which are the determination of the focus of expansion in 2-D, determination of the axis and direction of rotation and segmentation from motion. The sensor limitations were discussed extensively. Besides algorithmical constraints, the TED was found the determining factor for the sensor performance.

5.12 Future Work

Future work on the direction-of-motion sensor should concentrate on the following topics: Since the temporal edge detector is crucial for a reliable sensor operation it should be improved for very low and very high temporal contrasts. An adaptive TED threshold should be implemented such that for example the mean firing rate would be kept constant. In order to improve the algorithm, the facilitation time τ_{fac} and the persistence time τ_{pt} should be adaptive to the stimulus velocity. Both times should decrease for higher stimulus velocities, which would even increase the theoretical dynamic range of stimulus velocities and spatial frequencies. Even though the FTC algorithm was specifically developed for computing direction-of-motion in a very compact implementation, it could be desirable for some applications to obtain velocity information. The author has plans to extend the FTC algorithm to compute velocity in 2-D in a very compact implementation.

With the FTC sensor as a reliable front end device to compute the direction-of-motion field, two major research areas become interesting for further investigation. First the many possible uses of a direction-of-motion field of the kind the FTC sensor provides should be determined. Second the FTC sensor could be used in real world applications such as on a moving vehicle or in a oculomotor plant.

The investigation of the uses of the direction-of-motion field should be done with particular respect to the flexible real time averaging schemes of the on chip scanners. Any arbitrary combination of pixels can be summed together with the scanners, thus for example row averages and column averages can be obtained very easily in real time. Specifically algorithms for the following applications should be investigated:

- *Global translatory direction-of-motion.* Summing together the motion output of patches of pixels (pyramid like) increases the robustness of the result. Summing up all pixels of the array can yield very reliable direction-of-motion information possibly down to much lower contrasts than single pixels, also an increased angular resolution could be obtained.

- *Focus of expansion (FOE) and focus of contraction in 2-D.* Focus of expansion sensors have been built in analog VLSI hardware for 1-D (see [IKK95]). The author had the opportunity to test one of the sensors in real world, with dissatisfying result. With the FTC sensor and some additional circuitry determining the FOE in 2-D may be possible. Scenes that contain textured and non textured parts have to be taken into account and the effects on results obtained through averaging have to be considered.

- *Axis and direction of rotation.* Examples of rotating stimuli have been demonstrated in this work. For those stimuli determining the axis and direction of rotation may be possible through smart averaging.

- *Segmentation from motion.* Also demonstrated in this work are flow fields for objects moving in different directions. For only one moving object the row and column averages of the motion vectors can tell about its location and direction-of-motion. For two objects the binding problem has to be solved in order to determine, which 1-D average belongs to which object. Other segmentation from motion schemes may be possible to implement.

- *Sensor fusion of motion and gray value information.* On the FTC sensor the motion vector is locally available as well as the local image intensity. It might be favorable for segmentation tasks to combine those two cues for a more robust performance.

- *Fusion of multiple direction-of-motion fields.* Interesting experiments can be thought of with using multiple direction-of-motion sensors. For example with two FTC sensors spaced apart and orthogonally oriented the direction-of-motion of an object in 3-D could be determined.

- *Object tracking.* The use of analog VLSI sensors for object tracking can be advantageous over a serial computer implementation because the information is processed in parallel and the object location, once detected, can actively be reported. Using a direction-of-motion sensor for object tracking may improve the performance of the tracking system if the direction-of-motion signal can be used for predicting of the object path.

Chapter 6

Conclusions

In this work three different analog VLSI motion sensors have been described and characterized. All three sensors were proven to be fully functional. The CMotion1d sensor, the first working sensor to implement the gradient method in 1-D, was shown to compute the 1-D stimulus velocity in real time. It has been shown that due to the discrete implementation the velocity output becomes dependent on spatial frequency. The Gradient2d sensor uses a gradient based algorithm which allows for a very compact implementation in analog hardware. The combination of small pixel size in a 2-D implementation, desirable motion output for 2-D stimuli and high sensitivity makes this sensor exceptional. Finally the FTC sensor, which implements a feature based algorithm, was able to generate rather complex flow fields with just direction-of-motion information.

All three sensors share in common that they combine imaging and computation on one single chip. The entire motion detection system therefore consists of only the sensor chip, a lens, some potentiometers to supply the bias voltages and an operational amplifier to amplify the sensor output currents. A conclusive comparison between these sensors is not possible without a specific application in mind. Generally all of the sensors could be used where compactness, low mass, low power consumption, high speed, robustness against mechanical stress or a low price is needed. This could be in mobile robotics, in cars, in portable electronics and real time applications.

Analog VLSI motion sensors of the kind presented in this work could also be used in a very different context. Computer software models of biological vision systems can be very flexible and have been developed extensively in the past. For complex models, though, simulation times can become long. Additionally natural images are not usually used as input to the simulations. Analog VLSI implementations of models of vision systems could be used favorably in this context, because their operation is in real time, natural images can be captured, and their architecture is inherently parallel already. In particular with the motion sensors described in this thesis not only the motion information is available, but also the input image, so that sensory fusion might be investigated.

In the following the individual strengths and weaknesses of the sensors are summarized, and possible applications are proposed.

The **CMotion1d sensor** and a possible 2-D implementation of the gradient method are fascinating demonstrations of how mathematical operations can be performed in analog VLSI. An algorithm has been derived that could equally well be used in a machine vision computer application, and was straightforwardly implemented in hardware. This approach yielded a sensor which is capable of computing the true stimulus velocity, but at the cost of a larger pixel. The strengths of the CMotion1d sensor are:

- Continuous time velocity computation, as compared to sample-and-hold type sensors
- Velocity output independent of contrast down to 20% contrast
- Direction selectivity maintained down to 4 % contrast
- Single pixel velocity output approximately linear to stimulus velocity over a range of three orders of magnitude (measured from 0.072 mm/sec to 76 mm/sec)
- Velocity output almost linear to rotational velocity up to measured stimulus speeds of 353 rpm

whereas the following undesired properties of the CMotion1d sensor have been found:

- Large pixel size (147μm$\times$270μm in 2.0μm technology for 1-D sensor, 292μm$\times$292μm in 2.0μm technology for 2-D sensor)
- Spatial frequency dependence (predicted by the gradient method for a discrete spatial derivative)
- Divergent velocity output for stimulus directions close to $\pm 90^\circ$ (predicted by the gradient method, not due to the hardware implementation)

The CMotion1d sensor is suited for applications where a velocity signal is required, high speeds can occur and the input signal is approximately 1-D.

The **Gradient2d sensor** has been proven to be the most robust of all three presented sensors. Few biases are required and none of them is overly sensitive. The pixel size is very small due to the algorithm, which was developed specifically for a hardware implementation. In summary the advantages of this sensor are:

- Very small pixel size (217μm$\times$210μm in 2.0μm technology, 112μm$\times$112μm in 1.2μm technology)
- Motion output monotonic with stimulus velocity (measured from 0.05 mm/sec to 76 mm/sec)
- High sensitivity to low contrast stimuli. Direction selectivity maintained down to 2 % contrast
- Desirable orientation tuning curve, suited for 2-D operation (cosine shaped for not too high contrast stimuli)
- Few bias voltages required. No critical adjustments required

The following features could be seen as drawbacks of the Gradient2d sensor

- Time dependent motion output. If a more stable motion output is desired, it can be averaged over a short time

- Motion output dependent on contrast and spatial frequency. The dependency is such that the motion output degrades gracefully towards small contrasts and spatial frequencies and towards the aliasing point

The Gradient2d sensor is the most promising sensor if a high density array of direction of motion cells is desired. The sensor not only reports direction-of-motion continuously in angular space, but the length of the motion vectors additionally codes for the stimulus speed.

The **FTC sensor** was shown to compute rather complex direction-of-motion flow fields on a 12×13 pixel array. The motion output of every pixel is held on chip for a programmable time, which can be useful for some applications. The FTC sensor features the following properties:

- Small pixel size (119μm$\times$128μm in 1.2μm technology)
- Direction-of-motion detected correctly over a wide velocity range (90 % confidence for high contrast stimuli for speeds from 0.5 pixels/sec to 500 pixels/sec)
- Good contrast sensitivity (90 % confidence down to 20 % contrast)
- Programmable persistence time

Problem areas with the FTC sensor are

- Angular resolution only $\pm 45^o$
- Temporal edge detector is crucial for sensor performance (sensitivity, spike length and independency of stimulus shape)
- Critical biases in the temporal edge detector (threshold, TED bias and gain)

The FTC sensor might be used favorably for problem classes, where only the direction of motion field is required. Scalar parameters, like the focus-of-expansion, the focus-of-contraction, the axis-of-rotation, the direction of rotation, the overall direction-of-motion could be recovered from a direction-of-motion field. Additionally the FTC sensor might be used in sensory motor systems, where a feedback signal is required, that is derived from a moving scene. The FTC scheme could be improved to compute velocity, which would make this sensor even more versatile.

In summary, with the motion sensors presented in this work it has been proven, that (a) both correlation based *and* gradient based methods for motion computation can successfully be implemented in analog VLSI, and (b) high density 2-D motion sensors can be built.

Chapter 7

Zusammenfassung

In dieser Arbeit wurden drei neuartige Bewegungsdetektoren entwickelt, in analog VLSI Technologie implementiert, beschrieben und charakterisiert. Jeder der Sensoren ist vollständig funktionstüchtig.

Mit dem sog. ‘CMotion1d Sensor’ ist erstmals das Gradientenverfahren zur Geschwindigkeitsbestimmung erfolgreich implementiert worden. Der Sensor berechnet in Echtzeit in einer räumlichen Dimension die lokale Geschwindigkeit. Es wurde gezeigt, daß aufgrund der diskreten Ortsableitung das berechnete Geschwindigkeitssignal abhängig wird von dem Ortsfrequenzgehalt des jeweiligen Stimulus. Der sog. ‘Gradient2d Sensor’ benutzt ein anderes gradientenbasiertes Verfahren zur Berechnung eines zweidimensionalen Bewegungsvektorfeldes in Echtzeit. Die Abhängigkeit der Vektoren von der Geschwindigkeit, dem Kontrast, dem Ortsfrequenzgehalt und der Orientierung des Stimulus wurde untersucht und erklärt. Der Algorithmus erlaubt eine sehr kompakte Implementierung in analog VLSI Hardware. Die Kombination von geringem Flächenbedarf pro Pixel, gutem Verhalten für 2-D Stimuli und hoher Empfindlichkeit zeichnen diesen Sensor besonders aus. Schließlich wurde für den sog. ‘FTC Sensor’ ein neues korrelationsbasiertes Verfahren zur Bestimmung der Bewegungsrichtung entwickelt und implementiert. Recht komplexe Flußfelder können von diesem 2-D Sensor in Echtzeit berechnet werden.

Alle drei Sensoren haben gemeinsam, daß Bilderfassung und Bewegungsberechnung auf dem selben Chip geschehen. Dies hat den Vorteil, daß das komplette System zur Bewegungsberechnung nur aus sehr wenigen Komponenten besteht: Dem Sensorchip, einer Linse, einigen Potentiometern zur Erzeugung der Versorgungsspannungen, und einem Operationsverstärker zur Verstärkung der kleinen Ausgangsströme des Chips. Eine Entscheidung, welcher der Sensoren am besten sei, kann nicht gefällt werden, ohne daß dies im Hinblick auf eine bestimmte Anwendung des Sensors geschieht. Jeder der Sensoren kann verwandt werden für Anwendungen, bei welchen Kompaktheit, geringes Gewicht, geringer Energiebedarf, hohe Rechengeschwindigkeit oder Echtzeitverhalten, Robustheit gegen mechanische Beanspruchung und geringer Preis eine Rolle spielen. Dies ist zum Beispiel der Fall in Robotics Anwendungen, in der Automobiltechnik, in tragbaren Elektronikgeräten und in Echtzeitanwendungen.

Analog VLSI Sensoren von der Art, wie sie in dieser Arbeit vorgestellt werden, können auch in einer sehr verschiedenen Anwendung eingesetzt werden. Biologische visuelle Systeme sind in der Vergangenheit intensiv untersucht worden, oft mit Hilfe von detaillierten Computersimulationen. Für komplexe Modelle kann der Zeitbedarf für die Berechnungen sehr lange werden. Zusätzlich wird selten mit natürlichen Bildsequenzen gearbeitet. Hier bietet sich der Einsatz von analog VLSI Implementationen von biologischen Modellen an. Zum einen arbeiten diese Hardwaremodelle in Echtzeit und ihre Architektur ist parallel organisiert, zum anderen können in Echtzeit die Bilder verarbeitet werden, die auf dem selben Chip eingefangen werden. Insbesondere mit den Sensoren, welche in dieser Arbeit vorgestellt werden, kann sogar versucht werden, verschiedene Modalitäten wie Lichtintensität und Bewegung zu vereinigen.

Im Folgenden werden individuell für jeden Sensor die Vor-und Nachteile aufgeführt.

Der **CMotion1d Sensor** und eine mögliche Erweiterung des Gradientenverfahrens zur Geschwindigkeitsbestimmung in zwei Dimensionen sind Beispiele dafür, wie mathematische Operationen in einer Art analogem parallelem Computer umgesetzt werden können. Das Gradientenverfahren liefert zwar ein echtes *Geschwindigkeits*feld, nimmt aber keine Rücksicht auf einfache Implementierbarkeit in Hardware. Der entwickelte Sensor liefert deshalb ein Signal linear zur Geschwindigkeit des Stimulus, hat aber ein etwas größeren Pixel als die anderen hier vorgestellten Sensoren. Die Stärken des CMotion1d Sensors sind:

- Geschwindigkeitsberechnung in Echtzeit und kontinuierlich in der Zeit, im Gegensatz zu Sensoren, welche das Geschwindigkeitssignal einmal berechnen und dann speichern
- Geschwindigkeitssignal unabhängig vom Kontrast des Stimulus für Kontraste oberhalb 20 %
- Bewegungsrichtungen werden erkannt für Kontraste oberhalb 4 %
- Geschwindigkeitssignal eines einzelnen Pixels annähernd linear zur Geschwindigkeit des Stimulus über einen Geschwindigkeitsbereich von mehr als drei Dekaden (gemessen wurde für Stimulus Geschwindigkeiten von 0.072 mm/sec bis 76 mm/sec)
- Geschwindigkeitssignal für Rotationsbewegung annähernd linear zu Stimulusgeschwindigkeiten bis zu 353 Umdrehungen/sec

während die folgenden negativen Eigenschaften gefunden wurden:

- Relativ großer Flächenbedarf pro Pixel (147μm$\times$270μm in 2.0μm Technologie für 1-D Sensoren, 292μm$\times$292μm in 2.0μm Technologie für 2-D Sensoren)
- Abhängigkeit vom Ortsfrequenzgehalt des Stimulus (dies wird theoretisch vorhergesagt für diskrete Ortsableitungen)
- Divergentes Geschwindigkeitssignal für Bewegungsrichtungen des Stimulus nahe an $\pm 90^\circ$ (dies ist ebenfalls erwartet für die eindimensionale Gradientenmethode mit zweidimensionalen Stimuli)

Der CMotion1d Sensor ist geeignet für Anwendungen, bei welchen ein Signal benötigt wird, welches proportional zur Stimulusgeschwindigkeit ist. Zusätzlich können sehr kleine und sehr große Geschwindigkeiten verarbeitet werden, welche mit konventionellen Systemen oft nicht mehr erkannt werden können. Die Bedingung an die Stimuli ist, daß sie annähernd eindimensional präsentiert werden.

Der **Gradient2d Sensor** ist der robusteste der vorgestellten Sensoren. Nur wenige Versorgungsspannungen werden benötigt, und keine von ihnen ist sehr empfindlich einzustellen. Der Flächenbedarf pro Pixel ist sehr gering, da ein Algorithmus speziell für die Hardwareimplementation entwickelt wurde. Zusammenfassend sind die Vorteile des Gradient2d Sensors:

- Sehr kleine Pixel (217μm$\times$210μm in 2.0μm Technologie, 112μm$\times$112μm in 1.2μm Technologie)
- Das Bewegungssignal ist monoton abhänging von der Stimulusgeschwindigkeit (gemessen wurde der Bereich von 0.05 mm/sec bis 76 mm/sec)
- Hohe Empfindlichkeit selbst für sehr geringe Kontraste. Die Bewegungsrichtung eines Stimulus kann eindeutig erkannt werden für Kontraste bis zu 2 %
- Das Geschwindigkeitssignal verhält sich proportional zum Kosinus des Winkels der Stimulusbewegungsrichtung, was sehr vorteilhaft für einen 2-D Sensor ist
- Nur wenige Versorgungsspannungen werden benötigt. Keine von ihnen ist sehr kritisch einzustellen

Die folgenden Eigenschaften können als Nachteile des Gradient2d Sensors für manche Anwendungen angesehen werden:

- Bewegungssignal zeitabhängig. Dies steht im Gegensatz zu beispielsweise dem CMotion1d Sensor, welcher ein Geschwindigkeitssignal liefert, welches während der Stimuluspresentation annähernd konstant ist. Eine Zeitmittelung kann benutzt werden, um ein konstanteres Signal zu erzeugen.
- Bewegungssignal abhängig von Kontrast und Ortsfrequenzgehalt des Stimulus. Diese Abhängigkeit ist ‘gutmütig’ in dem Sinne, daß für kleine Kontraste und Frequenzen nahe Null und nahe der Aliasingfrequenz das Bewegungssignal verschwindet und nicht divergiert.

Der Gradient2d Sensor ist der vielversprechendste Sensor für Anwendungen, bei welchen ein hochauflösendes Vektorfeld und robuste, einfache Handhabbarkeit erwünscht sind. Der Gradient2d Sensor erzeugt nicht nur ein Bewegungsrichtungsvektorfeld in Echtzeit, sondern die relative Länge der Vektoren hat eine Aussage über die Geschwindigkeit der bewegten Objekte.

Der **FTC Sensor** konnte trotz der Einschränkung, daß nur die Bewegungs*richtung* und nicht die Geschwindigkeit berechnet werden, auf einem 13 $\times$ 12 Pixel großen Sensor recht komplexe Flußfelder erzeugen. Das Bewegungssignal eines jeden Pixel wird auf dem Chip für eine programmierbare Zeit lange gespeichert, und kann während dieser Zeit abgerufen werden. Der FTC Sensor hat die folgenden Vorzüge:

- Geringer Flächenbedarf pro Pixel (119μm$\times$128μm in 1.2μm Technologie)
- Bewegungsrichtung wird richtig erkannt über einen weiten Bereich von Stimulusgeschwindigkeiten (90 % Zuverläßigkeit für Stimuli mit starken Kontrasten für Geschwindigkeiten von 0.5 Pixel/sec bis 500 Pixel/sec)
- Gute Kontrastempfindlichkeit (90 % Zuverläßigkeit bis zu Kontrasten von 20 %)

- Programmierbare Speicherzeit des Bewegungssignales auf dem Sensor

Problemgebiete des FTC Sensors sind

- Winkelauflösung nur $\pm 45^o$
- Die Verläßlichkeit des Merkmalsdetektors ('TED') ist ausschlaggebend für die Leistungsfähigkeit des FTC Sensors. Wenn der TED Merkmale in der visuellen Szene nicht erfasst, dann kann kein Bewegungssignal berechnet werden
- Manche Versorgungsspannungen des TED müssen genau eingestellt werden

Der FTC Sensor kann vorteilhaft für Aufgabengebiete eingesetzt werden, für welche ein Bewegungsrichtungsfeld ausreichend und die Geschwindigkeitsinformation nicht notwendig ist. Skalare Parameter, wie zum Beispiel der Fluchtpunkt, die Lage der Rotationsachse, die Richtung der Rotation und die globale Bewegungsrichtung könnten von einem Bewegungsrichtungsfeld mit dem FTC Sensor in Echtzeit abgeleitet werden. Zusätzlich könnte der FTC Sensor in Sensor-Motor Systemen eingesetzt werden, um ein Rückkopplungssignal zu erzeugen. Der Algorithmus des FTC Sensors könnte erweitert werden, so daß auch die Geschwindigkeit berechnet werden kann. In diesem Falle wären die Einsatzgebiete noch vielfältiger.

Zusammenfassend wurde mit dieser Arbeit gezeigt, daß (a) sowohl korrelations- als auch gradientenbasierte Verfahren zur Bewegungsberechnung erfolgreich in analog VLSI Hardware implementiert werden können, und (b) hochauflösende 2-D Bewegungssensoren in analog VLSI möglich sind.

Appendix A

Mathematical Derivations

A.1 Mean output of the Gradient2d sensor

The mean expected motion output of the Gradient2d sensor is derived for a sine wave stimulus

$$I(x,t) = I_o \sin(\omega\, t - k\, x) \tag{A.1}$$

where $v = \omega/k$ is its velocity and k its spatial frequency.

The theoretical motion output of the Gradient2d sensor was given in equation 4.1 and is for this particular stimulus and a discrete spatial derivative evaluated at a pixel at $x = 0$

$$\begin{aligned} u(x=0,t) &= u_o\, I_t\, \tanh(I_\Delta\, \lambda) = \\ &= u_o\, I_o\, \omega\, \cos(\omega\, t)\, \tanh(I_o\, k\, \lambda\, \frac{\sin(k\,\Delta)}{k\,\Delta}\, \cos(\omega\, t)) \end{aligned} \tag{A.2}$$

and the motion output has to be considered only in one spatial dimension. Δ is the pixel spacing.

The mean motion output for is therefore given by

$$\begin{aligned} \bar{u}^t &= \frac{1}{T}\int_0^T u(t)\, dt = \\ &= v\, \frac{u_o}{\lambda} \sum_{n=1}^{\infty} a_n \left(I_o\, k\, \lambda \frac{\sin(k\,\Delta)}{k\,\Delta}\right)^{2\,n} \frac{1}{T}\int_0^T \cos^{2\,n} \omega\, t\, dt \end{aligned} \tag{A.3}$$

where the a_n are the Taylor coefficients of the tanh series. The time average cosine integral is given by

$$\frac{1}{T}\int_0^T \cos^{2\,n}\omega\, t\, dt = \sum_{i=1}^{n} \prod_{j=1}^{i} \frac{2\,j-1}{2\,j} \tag{A.4}$$

It can be seen that the expected mean velocity output is linear in the stimulus velocity for a given fixed spatial frequency k. The first two terms of the time average are:

$$\bar{u}_2^t = v\, \frac{u_o}{\lambda} \left[\frac{1}{2} \left(I_o\, k\, \lambda\, \frac{\sin(k\,\Delta)}{k\,\Delta}\right)^2 - \frac{1}{8} \left(I_o\, k\, \lambda\, \frac{\sin(k\,\Delta)}{k\,\Delta}\right)^4 \right] \tag{A.5}$$

Bibliography

[AB96] A. Andreou and K.A. Boahen. Translinear circuits in subthreshold MOS. *Analog Integrated Circuits and Signal Processing*, 9:141–166, 1996.

[ABP^{+}91] A.G. Andreou, K. A. Boahen, P.O. Pouliquen, A. Pavasovic, R. E. Jenkins, and K. Strohbehn. Current-mode subthreshold MOS circuits for analog VLSI neural systems. *IEEE Transactions on neural networks*, 2:205–213, 1991.

[BD97] C. Born and R. Deutschmann. Measurement of fast rotation by VLSI circuits. *Proceedings DAGM97, in press*, 1997.

[BDK97] C. Born, R. A. Deutschmann, and Christof Koch. Real time ego-motion estimation with neuromorphic analog VLSI sensors. *Proc. of the 4th Joint Symposium on Neural Computation, University of Southern California, Los Angeles, in press*, 1997.

[BFB92] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow techniques. Technical report no.299, Dept. of Computer Science, Univ. of Western Ontario, 1992.

[Boa96a] K.A. Boahen. Retinomorphic vision systems. *MicroNeuro'96: fifth int. conf. on neural networks and fuzzy systems*, 1996.

[Boa96b] K.A. Boahen. Retinomorphic vision systems I: Pixel design. *Proc. IEEE Int. Symp. on Circuits and Systems*, 1996.

[Boa96c] K.A. Boahen. Retinomorphic vision systems II: Communication channel design. *Proc. IEEE Int. Symp. on Circuits and Systems*, 1996.

[Boh94] S. Bohrer. *Visuelle Hinderniserkennung durch Auswertung des optischen Flusses in inversperspektivischen Szenen*. VDI Verlag, Reihe 8, 1994.

[Del93] T. Delbrück. Silicon retina with correlation-based, velocity-tuned pixels. *IEEE Trans. on Neural Networks*, 4:529–541, 1993.

[DeW92] S.P. DeWeerth. Analog VLSI circuits for stimulus localization and centroid computation. *Intern. Journal of Comp. Vision*, 8:191–202, 1992.

[DHK97a] R. A. Deutschmann, C. Higgins, and C. Koch. Neuromorphic analog VLSI sensors for 2-D direction of motion. *Proc. of the 4th Joint Symposium on Neural Computation, University of Southern California, Los Angeles, in press*, 1997.

[DHK97b] R. A. Deutschmann, C. Higgins, and C. Koch. Real-time analog VLSI sensors for 2-D direction of motion. In *Artificial Neural Networks - ICANN'97*, volume 1327 of *Lecture Notes in Computer Science*, pages 1163–1168. Springer Verlag, 1997.

[Dic95] E. Dickmanns. Road vehicle eyes for high precision navigation. *3rd Intl. Workshop on High Precision Navigation*, April 1995.

[DM91] T. Delbrück and C. Mead. Time-derivative adaptive silicon photoreceptor array. *SPIE Infrared Sensors: Detectors, Electronics, and Signal Processing*, 1541:92–99, 1991.

[DM94] T. Delbrück and C. Mead. Analog VLSI phototransduction. *CNS Memo No.30, Caltech*, 30:139–161, 1994.

[ECdSM97] R. Etienne-Cummings, J. Van der Spiegel, and P. Mueller. A focal plane visual motion measurement sensor. *IEEE Trans. Circuits and Systems 1*, 44:55–66, 1997.

[GM93] P. R. Gray and R. G. Meyer. *Analysis and design of analog integrated circuits*. John Wiley and Sons, 3 edition, 1993.

[HBB$^+$92] T.K. Horiuchi, W. Bair, B. Bishofberger, A. Moore, and C. Koch. Computing motion using analog VLSI vision chips: an experimental comparison among different approaches. *Intern. Journal of Computer Vision*, 8:203–216, 1992.

[HK97a] R. R. Harrison and C. Koch. A current-mode analog VLSI implementation of a fly elementary motion detector. *Proc. of the 4th Joint Symposium on Neural Computation, University of Southern California, Los Angeles, in press*, 1997.

[HK97b] C.M. Higgins and C. Koch. Analog CMOS velocity sensors. *Proceedings of Electronic Imaging '97 SPIE*, 3019, 1997.

[HKL90] J.G. Harris, C. Koch, and J. Luo. A 2-dimensional analog VLSI circuit for detecting discontinuities in early vision. *Science*, 248:1209–1211, 1990.

[HMKD97] T.K. Horiuchi, T.G. Morris, C. Koch, and S.P. DeWeerth. Analog VLSI circuits for attention-based, visual tracking. In *Advances in Neural Processing Systems 9*. MIT press, 1997. in press.

[Hor97] T. Horiuchi. *Analog VLSI-Based, Neuromorphic Sensorimotor Systems: Modeling the Primate Oculomotor System*. PhD thesis, California Institute of Technology, 1997.

[HS81] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[IKK95] G. Indiveri, J. Kramer, and C. Koch. Analog VLSI architecture for computing heading direction. *Proc. Intelligent Vehicles 1995*, 1995.

[Kra96] J. Kramer. Compact integrated motion sensor with three-pixel interaction. *IEEE Trans. Pattern Anal. Machine Intell.*, 18:455–560, 1996.

[KSK97] J. Kramer, R. Sarpeshkar, and C. Koch. Pulse-based analog VLSI velocity sensors. *IEEE Trans. Circuits and Systems II*, 44:86–101, 1997.

[LH82] R.F. Lyon and M.P. Haeberli. Designing and testing the optical mouse. *VLSI design*, pages 20–29, 1982.

[MD91a] C. A. Mead and T. Delbrück. Scanners for visualizing activity of analog VLSI circuitry. *Analog Integrated Circuits and Signal Processing*, 1:93–106, 1991.

[MD91b] C. A. Mead and T. Delbrück. Scanners for visualizing activity of analog VLSI circuitry. *Caltech Computation and Neural Systems Memo Number 11*, 1991.

[MDHM96] B.A. Minch, C. Diorio, P. Hasler, and C. Mead. Translinear circuits using subthreshold floating-gate MOS-transistors. *Analog Integrated Circuits and Signal Processing*, 9:167–179, 1996.

[ME83] J. Maunsell and D. Van Essen. Functional properties of neurons in middle temporal visual area of the Macaque monkey. I selectivity for stimulus direction, speed and orientation. *J. Neurophysiology*, 49:1127–1147, 1983.

[Mea89] C. Mead. *Analog VLSI and Neural Systems*. Addison-Wesley, 1989.

[MK91] A. Moore and C. Koch. A multiplication based analog motion detection chip. *Proc. SPIE, Visual Information processing: From neurons to chips*, 1473:66–75, 1991.

[MOS] MOSIS web page. http://www.mosis.org/info/domestic-price-list.inf.

[Nak85] K. Nakayama. Biological image motion processing: a review. *Vision Research*, 25:625–660, 1985.

[Opp83] A. Oppenheim. *Signals and Systems*. Prentice-Hall, 1983.

[Pap84] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, second edition, 1984.

[PAW94] A. Pavasovic, A. Andreou, and C.R. Westgate. Characterization of subthreshold MOS mismatch in transistors for VLSI systems. *Analog Integrated Circuits and Signal Processing*, 6:75–85, 1994.

[Rei87] W. Reichardt. Evaluation of optical motion information by movement detectors. *Journal Comp. Phys. A*, 161:533–547, 1987.

[SBK93] R. Sarpeshkar, W. Bair, and C. Koch. Visual motion computation in analog VLSI using pulses. In S. Hanson, J. Cowan, and C. Giles, editors, *Neural Information Processing Systems 5*, pages 781–788, 1993.

[Sze81] S. M. Sze. *Physics of semiconductor devices*. A Wiley-Interscience publication, 2 edition, 1981.

[TM86] J. Tanner and C. Mead. An integrated optical motion sensor. *VLSI Signal Processing II, IEEE Press*, pages 59–76, 1986.

[VGT90] A. Verri, F. Girosi, and V. Torre. Differential techniques for optical flow. *J. Opt. Soc. Am. A*, 7:912–922, 1990.

[WE85] N. Weste and K. Eshraghian. *Principles of CMOS VLSI design*. Addison-Wesley, 1985.

Acknowledgments

I wish to thank the following people:
Annike, my soon to be wife, for the love, support and freedom she gave me; my parents; Timothy 'Timmer' Horiuchi for teaching me the art of analog VLSI, for numerous coffees and being a good friend; Christof Koch, my advisor at Caltech, whose enthusiasm is fantastic; Reid 'Flyboy' Harrison for his company during long nights in the lab and spirited discussions; Charles 'Chuck' Higgins for an enjoyable collaboration; Prof. Van Hemmen, my advisor at TU Munich; Christof Born for proofreading this thesis; Martin Stemmler and Amit Manwani.
I gratefully acknowledge the support of the Studienstiftung des deutschen Volkes.