

ABSTRACT

Title of dissertation: **BIO-INSPIRED VLSI SYSTEMS:
FROM SYNAPSE TO BEHAVIOR**

Peng Xu, Doctor of Philosophy, 2008

Dissertation directed by: Professor Pamela Abshire
Department of Electrical and Computer
Engineering

We investigate VLSI systems using biological computational principles. The elegance of biological systems throughout the structure levels provides possible solutions to many engineering challenges. Specifically, we investigate neural systems at the synaptic level and at the sensorimotor integration level, which inspire our similar implementations in silicon. For both VLSI systems, we use floating gate MOSFETs in standard CMOS processes as nonvolatile storage elements, which enable adaptation and programmability.

We propose a compact silicon stochastic synapse and methods to incorporate activity-dependent dynamics, which emulate a biological stochastic synapse. It takes advantage of the ubiquitous noise in circuits and implements computation in the spike domain. We implement and demonstrate the silicon stochastic synapse with short-term depression by modulating the influence of noise on the circuit. Such synapse has not been reported in the literature. The circuit exhibits true randomness and similar behavior of rate normalization and information redundancy reduction

as its biological counterparts. The circuit behavior also agrees well with the theory and simulation of a short-term depression circuit model based on a subtractive single release model.

To understand the stochastic behavior of the silicon stochastic synapse and the stochastic operation of conventional circuits due to semiconductor technology scaling, we develop the stochastic modeling of circuits. Transient analysis from the numerical solution of the stochastic model provides the sample path and ensemble statistics. The analytical solution of steady state distribution could be obtained from first principles. Small signal stochastic models show the interaction between noise and circuit dynamics, elucidating the effect of device parameters and biases on the stochastic behavior.

We investigate optic flow wide field integration based navigation inspired from the fly in simulation, theory, and VLSI design. We generalize the framework to limited view angles. We design and test an integrated motion image sensor with on-chip optic flow estimation, adaptation, and programmable spatial filtering to directly interface with actuators for autonomous navigation. This is the first reported image sensor that uses the spatial motion pattern to extract motion parameters enabled by the mismatch compensation and programmable filters. It provides light weight and low power integrated approach to autonomous navigation of micro air vehicles. The mismatch compensation and programmable filters can also be applied to other sensory front-end where on-chip spatial processing is required and distortion from fabrication mismatch among sensor units has to be reduced. The sensor is integrated with a ground vehicle and navigation through simple tunnel environments

is demonstrated with limited information from only one horizontal line of optical input of height 2.4° and field of view angle 83.1° .

BIO-INSPIRED VLSI SYSTEMS:
FROM SYNAPSE TO BEHAVIOR

by

Peng Xu

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2008

Advisory Committee:
Professor Pamela Abshire, Chair/Advisor
Professor P. S. Krishnaprasad
Professor Robert Newcomb
Professor S. K. Gupta
Professor J. Sean Humbert

© Copyright by
Peng Xu
2008

Dedicated to the memory of my beloved sister, Xin Xu.

Acknowledgments

First I would like to thank my parents for their cultivation of my strong curiosity, their endless support of my pursuit of knowledge, and their sacrifice for my education. I would like to thank my wife, Tao, who always gives me a peace of mind.

Prof. Pamela Abshire is not only my advisor but also my friend. I know she is always there to help me. I thank her for giving me the freedom, the guidance, the encouragement, and the financial support to explore and pursue my research interests. The warm feeling is precious.

Prof. Krishnaprasad, who set a role model for me as a professor, always has his door open to the students and always greets me "how is the world treating you?" I got my unofficial education of control from sitting in all his classes. I thank Prof. Horiuchi, popularly known as Timmer, for many inspiring discussions. I cannot remember how many times a supposed five minute discussion became an hour long. I thank Prof. Robert Newcomb, Prof. S. K. Gupta, and Prof. Sean Humbert for serving in my committee from their busy schedule.

I like to thank my labmates in the Integrated Biomorphic Information Systems Lab. I am fortunate to have such a harmonious lab environment. Somashekar Bangalore offered many suggestions on data acquisition. Marc P. Dandin was always there when I had all-nighters and ready to save me when I passed out. Alfred Haas and I shared the same craziness to build silicon brain. Nicole Nelson kindly supported everyone as the de facto lab manager. David Sander is my tennis buddy.

Anshu Sarje, Babak Nouri, Timir Datta are the new blood of the lab. Yiming Zhai helped me in Hspice simulation. Yanyi(Eric) Wong, Honghao Ji, and I had many discussions about circuits and I followed their steps to be the No. 3 of the lab.

I want to thank Prof. Sean Humbert, Mike Chinn, Andrew Hyslop, Joseph Conroy, and Kedar D. Dimble from the Department of Aerospace Engineering for a wonderful collaboration. I want to thank Thanos Chryssis for his help on setting up the view angle response experiment. I want to thank Shyam Mehrotra and Jay Renner for lending me the equipment and handling my chip submission.

I enjoyed working as a teaching assistant with Prof. Eyad Abed, Pamela Abshire, and John Melngailis. They showed me how to be an excellent teacher. Thank John for his confidence and trust in me, allowing me to teach part of the course.

I like to thank the faculty here at the University of Maryland for their dedication to teaching and research, which set the high standard for their students. Special thanks to Prof. Christopher Davis, Mario Dagenais, Edward Ott, Martin Peckerar, Andre Tits, and Todd Troyer. I also want to thank the people in the business office, graduate studies office, and information technology office, specially Mandi Elwood, Marion Devaney, Kristin Little, Ronald Jean, Maria Hoo, Vivian Lu, Tracy Chung, Peg Jayant, and Carlos Luceno, for letting me have on time orders, smooth conference trips, hassle free computer systems, and enjoyable graduate life.

I want to thank the neuromorphic workshop at Telluride 2005. It was such an inspiring experience that fuelled my research.

I thank MOSIS and NSF for providing chip fabrication services and financial support.

Finally I thank all the people I didn't name here for their help, encouragement, and trust.

Table of Contents

List of Tables	ix
List of Figures	x
List of Abbreviations	xviii
1 Preamble	1
1.1 Introduction	1
1.2 Organization	10
1.3 Contributions	12
1.3.1 Dynamic VLSI Stochastic Synapse	12
1.3.2 Stochastic Circuit Modeling	13
1.3.3 Fly Inspired Integrated Approach to Autonomous Navigation .	13
2 Silicon Stochastic Synapse with Plasticity	15
2.1 Introduction	15
2.1.1 Neuron and Synapse	15
2.1.2 Synaptic Plasticity	18
2.1.3 Adaptive VLSI Synapse	20
2.1.4 Randomness in VLSI	22
2.2 Silicon Stochastic Synapse	24
2.3 Stochastic Synapse with Short-Term Depression	29
2.3.1 Short-term Depression Circuit Model	29
2.3.2 Simulation	31
2.3.3 Circuit Implementation	33
2.4 Experimental Results	40
2.4.1 Stochastic Synapse Tested as RNG	42
2.4.1.1 Statistical Tests	42
2.4.1.2 Adjustable Probability	47
2.4.1.3 Interference	48
2.4.2 First Generation Stochastic Synapse with STD	49
2.4.3 Second Generation Stochastic Synapse with STD	53
3 Stochastic Circuit Modeling and Simulation	61
3.1 Introduction	61
3.2 Circuit Noise	62
3.2.1 Thermal Noise	63
3.2.2 Shot Noise	64
3.2.3 Flicker Noise	64
3.2.4 Burst Noise	65
3.2.5 Avalanche Noise	65
3.3 Stochastic Differential Equations	65
3.3.1 Random Variable and Stochastic Process	66

3.3.2	Stochastic Integrals	68
3.3.2.1	Integral $J(f)(\omega) = \int_a^b f(s, \omega) ds$	68
3.3.2.2	Ito Integral $I(f)(\omega) = \int_a^b f(s, \omega) dW(s, \omega)$	69
3.3.3	Stochastic Differential Equation	70
3.3.4	Numerical Solution of Stochastic Differential Equation	71
3.3.5	Fokker-Planck Equation	73
3.4	Stochastic Circuit Model	73
3.4.1	Voltage Node Stochastic Model	73
3.4.2	Stochastic Synapse Circuit	75
3.4.2.1	Stochastic Model of the Stochastic Synapse Circuit	75
3.4.2.2	Small Signal Stochastic Model	80
3.4.3	CMOS Inverter	85
3.4.3.1	Dynamics of an Ideal CMOS Inverter	86
3.4.3.2	Stochastic Model of a CMOS Inverter	88
3.4.3.3	Numerical Simulation of the CMOS Inverter	89
3.4.3.4	Analytic Steady State Solution	91
4	Fly Inspired Autonomous Navigation: Theory and Simulation	94
4.1	Background	94
4.1.1	The Visual System of the Fly	94
4.1.2	The Sensorimotor System of the Fly	96
4.1.3	Visual Navigation in Flying Insects	97
4.1.4	Natural Environment	99
4.2	Optic Flow Wide Field Integration (WFI) based Navigation	100
4.2.1	WFI based Navigation	101
4.2.2	Elementary Motion Detector	102
4.2.3	EMD in WFI based Navigation	104
4.3	Limited View Angle Solution	108
4.3.1	View Angle of π	109
4.3.2	General Case	115
5	Motion Image Sensor with On-chip Adaptation and Programmable Filtering	119
5.1	EMD Implementation and Mismatch	120
5.1.1	EMD Implementation	120
5.1.1.1	Photoreceptor	121
5.1.1.2	GmC Type Filter	123
5.1.1.3	Gilbert Multiplier [1]	124
5.1.2	Mismatch	125
5.2	On-chip Spatial Filter	125
5.2.1	Resistive Networks based Implementation	127
5.2.1.1	Sinusoidal Function Circuit	128
5.2.1.2	Sinusoidal Spatial Filter	130
5.2.2	Nonvolatile Storage based Approach	131
5.2.2.1	P-type Floating Gate MOSFET	133

5.2.2.2	Basic Mechanism for Mismatch Compensation and Filter Programming	136
5.2.2.3	Programmable Current Element (PCE)	139
5.3	Motion Image Sensor Architecture and Operation	144
5.3.1	Programmable Current Matrix (PCM)	146
5.3.2	Column Control Logic	148
5.3.3	Filter Select Control Logic	149
5.3.4	Operation Mode	149
5.3.4.1	Filter Programming	151
5.3.4.2	Mismatch Compensation	151
5.3.4.3	Sensing and Filtering	151
5.4	Experimental Results	152
5.4.1	PCE Test	153
5.4.2	System Integration and Test	164
5.4.2.1	System Integration	164
5.4.2.2	View Angle Characterization and Filter Programming	167
5.4.2.3	Open Loop Test	168
5.4.2.4	Closed Loop Test	170
6	Conclusions and Open Problems	176
A	Adaptive Large Time Constant Filter	182
A.1	Introduction	182
A.2	Basic Structure	184
A.3	Operation Configuration	185
A.4	Simulation Results	185
B	Adaptive Floating Gate Pixel	190
B.1	Floating Gate Pixel Structure	190
B.2	Frequency Analysis	192
	Bibliography	196

List of Tables

2.1	Synaptic plasticity	20
2.2	NIST statistical test results	46
3.1	Comparison of δ in probability tuning	84
5.1	Statistics of the PCE output	161
5.2	Ratio of filter output during rotation	170
A.1	Cutoff frequency	187

List of Figures

2.1	Stochastic synapse circuit: (a) clocked cross-coupled differential pair comparator, (b) dynamic output buffer, (c) input-output behavior. $\frac{W_{1,2}}{L_{1,2}} = \frac{14}{2}, \frac{4}{12}$ have been used.	25
2.2	Block diagram of silicon stochastic synapse with short-term plasticity.	28
2.3	Block diagram of silicon stochastic synapse with long-term plasticity.	28
2.4	Mean probability as a function of input spike rate for $\Delta v = 2, 4, 6$ mV. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The solid lines show the least mean square fit for input rates from 400 Hz to 1000 Hz. $\tau_d = 100$ ms.	31
2.5	Mean probability as a function of input spike rate for $\tau_d = 100, 200, 300$ ms. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The solid lines show the least mean square fit for input rates from 400 Hz to 1000 Hz. $\Delta v = 2$ mV.	32
2.6	Simulated probability trajectory over 200 ms period. $r = 100$ Hz, $\tau_d = 100$ ms, $\Delta v = 2$ mV.	32
2.7	Autocorrelation of output spike trains for different time constants τ_d , each spike interval is 10 ms.	34
2.8	Power spectral density of output spike trains for different time constants τ_d	35
2.9	Autocorrelation of output spike trains for different voltage drop Δv , each spike interval is 10 ms.	36
2.10	Block diagram of silicon stochastic synapse with STD.	37
2.11	Block diagram of pulse shaping circuit.	38
2.12	Transient simulation of V_{i-} driven by 5 transmitted spikes at 50 Hz, $V_t = 0.7$ V, $V_w = 0.4$ V, $V_h = 0.5$ V.	39
2.13	Schematic of a compact implementation of stochastic synapse with STD.	40
2.14	Autocorrelation of the bit sequence from one RNG (\times) and cross-correlation between bit sequences from two RNGs (\cdot). For clarity, only data up to a sample lag of 100 is shown.	43

2.15	Power spectral density of one bit sequence at sampling frequency 1 kHz (similar results are obtained for measurements up to 200 kHz).	44
2.16	Power spectral density of cross correlation between two random bit sequences.	44
2.17	Output probability $p(V_{o+} > V_{o-})$ as a function of input offset $v = V_{i+} - V_{i-}$, the solid line is $f(v) = 0.5 \left(1 + \operatorname{erf} \left(\frac{v-\mu}{\sqrt{2}\delta} \right) \right)$, where $\mu = 0.71$ and $\delta = 2.16$.	47
2.18	Autocorrelation of the output spike train from the stochastic synapse with STD for an input spike rate of 100 Hz. Autocorrelation at time zero represents the sequence variance. Negative autocorrelation at short time intervals indicates STD.	50
2.19	Power spectral density of the output spike train from the stochastic synapse with STD for an input spike rate of 100 Hz. Lower PSD at low frequencies indicates STD.	50
2.20	Mean probability as a function of the input spike rate. Data were collected at input rates of 50 Hz, 100 Hz, 200 Hz, 300 Hz, 400 Hz, 500 Hz, and 1000 Hz. The solid line shows the least mean square fit $p(x) = 18x - 3.7 \times 10^{-5}$.	51
2.21	Mean probability as a function of the input spike rate for various combinations of V_t and V_w . $V_r = 1.60$ V, and $V_{icm} = 2$ V. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit for input rates from 200 Hz to 1000 Hz.	52
2.22	Mean probability as a function of the input spike rate for $V_r = 1.55$ and 1.60 V. $V_t = 0.8$ V, $V_w = 0.3$ V, and $V_{icm} = 2$ V. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit for input rates from 200 Hz to 1000 Hz.	52
2.23	Autocorrelation of output spike trains from the silicon stochastic synapse with STD for an input spike rate of 100 Hz. Autocorrelation at time zero represents the sequence variance, and negative autocorrelation at short time intervals indicates STD.	54
2.24	Power spectral density of output spike trains from the silicon stochastic synapse with STD for an input spike rate of 100 Hz. Lower PSD at low frequencies indicates STD.	55

2.25	Characterization of the output spike train from the simulation of the stochastic synapse with STD. $r = 100$ Hz, $\tau_d = 220$ ms, $\Delta v = 5$ mV, $V_{max} = 5$ mV.	56
2.26	Mean probability as a function of input spike rate for pulse width $T_p = 10, 20, 30, 40, 50$ μs . Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit from 200 Hz to 1000 Hz.	57
2.27	$a\Delta v\tau_d$ as a function of the pulse width. The dotted line shows the least mean square fit, $f(x) = 0.0008x + 0.0017$	57
2.28	Mean probability as a function of the input spike rate for $V_r = 1.55, 1.56, 1.57, 1.58, 1.59$ V. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit.	58
2.29	$a\Delta v\tau_d$ as a function of V_r . The dotted line shows the least mean square fit, $f(x) = e^{(44.54x-72.87)}$	58
2.30	Mean probability as a function of the input spike rate for $V_w = 0.30, 0.35, 0.40, 0.45, 0.50$ V. The dotted lines show the least mean square fit for input rates from 200 Hz to 1000 Hz.	59
2.31	$a\Delta v\tau_d$ as a function of V_w . The dotted line shows the least mean square fit, $f(x) = e^{(15.47x-9.854)}$	60
3.1	Schematic of a simple voltage node.	73
3.2	Stochastic synapse circuits: a) the clocked cross-coupled differential pair comparator in the stochastic synapse circuit, b) the simplified circuit for stochastic modeling, parasitic capacitances are explicitly added.	75
3.3	One simulated sample path of the stochastic synapse, dashed line: V_{op} , solid line: V_{om}	77
3.4	Close-up view of the initial period of Fig. 3.3, dashed line: V_{op} , solid line: V_{om}	78
3.5	Multiple sample paths from 10 transient simulations, dashed line: V_{op} , solid line: V_{om}	78
3.6	The probability $p(V_{op} > V_{om})$ as a function of input offset $v = V_{ip} - V_{im}$. The solid line shows the fitting of an error function $f(v) = 0.5 \left(1 + \operatorname{erf} \left(\frac{v-\mu}{\sqrt{2}\delta} \right) \right)$, where $\mu = 0.0157$ and $\delta = 1.2659$	79

3.7	The probability $p(V_{op} > V_{om})$ as a function of input offset $V_{ip} - V_{im}$ for $I_{bias} = 100 \mu\text{A}, 10 \mu\text{A}, 1\mu\text{A}, 100 \text{ nA}, 10 \text{ nA}, 1 \text{ nA}, 100 \text{ pA},$ and 10 pA . $\delta=2.8039, 1.6547, 1.2659, 1.1666, 1.1361, 1.1402, 1.1216,$ and 1.1246 respectively, for the fitting of the error function.	79
3.8	Small signal stochastic model of the stochastic synapse circuit.	80
3.9	The probability $p(V_{op} > V_{om})$ as a function of input offset $V_{ip} - V_{im}$, \circ : from the above threshold small signal model (3.41) for $I_{bias} = 100 \mu\text{A}, 10 \mu\text{A},$ \times : from the subthreshold small signal model (3.40) for $I_{bias} = 10 \text{ nA}, 1 \text{ nA}, 100 \text{ pA}, 10 \text{ pA},$ and 1 pA . $\delta = 3.5990, 1.4475, 1.0776, 1.0980, 1.1098, 1.0972,$ and 1.0903 for the fitting of the error function.	83
3.10	A CMOS inverter with a capacitive load.	86
3.11	Output voltage as a function of time for the ideal CMOS inverter in subthreshold operation after switching at input, $V_{dd} = 0.1 \text{ V}$	88
3.12	Output voltage as a function of time for the ideal CMOS inverter in subthreshold operation after switching at input, $V_{dd} = 0.01 \text{ V}$	89
3.13	Output voltage as a function of time for the CMOS inverter with noise in subthreshold operation after input transition, 10 runs with $V_{dd} = 0.1 \text{ V}$	90
3.14	Output voltage as a function of time for the CMOS inverter with noise in subthreshold operation after switching at input, 10 runs with $V_{dd} = 0.01 \text{ V}$	90
3.15	The probability density function of output voltage for a CMOS inverter with noise in subthreshold operation, $V_{dd} = 0.1 \text{ V}$	91
4.1	Cartoon of fly's sensory motor systems.	97
4.2	Optic flow WFI based feedback control [2].	101
4.3	Reichardt EMD.	102
4.4	Elaborated EMD. P: photoreceptor, SF: spatial filter, BPF: band-pass filter, LPF: low-pass filter.	104
4.5	Velocity tuning from slow EMD (peak velocity = $10^\circ/s$), fast EMD (peak velocity = $50^\circ/s$), and normalized fast EMD.	106
4.6	EMD circular sensor.	106

4.7	Simulation of WFI based navigation with EMD sensors. (a)Navigation through a tunnel, (b)Four matched filter outputs from EMD based optic flow (dotted line) and ideal optic flow (solid line), a_0 : DC filter, a_1 : $\cos(\cdot)$ filter, a_2 : $\cos(2\cdot)$ filter, u_2 : torque filter.	107
4.8	Vehicle configuration in planar tunnel.	109
4.9	Simulation of WFI based navigation using limited view angle optic flow. a)View angle: π , b)View angle: $\pi/2$	114
5.1	Block diagram of EMD implementation.	120
5.2	Schematic of photoreceptor.	121
5.3	Block diagram of GmC filters: (a) high-pass filter, (b) low-pass filter.	123
5.4	Schematic of transconductance amplifier.	123
5.5	Schematic of Gilbert multiplier.	125
5.6	Steady state tuning curve of 10 EMDs, positive frequency: stimuli moving from right to left, negative frequency: stimuli moving from left to right.	126
5.7	Schematic of a sinusoidal function circuit.	128
5.8	Differential current output I_o of the $\sin(\cdot)$ function circuit for a DC sweep simulation of the input offset V_{offset} . The theoretical approximation is $5.61 \sin(x)$ with $\kappa = 0.65$. The scaled theoretical approximation is $6.06 \sin(x)$, obtained by scaling $\sin(x)$ with the magnitude of the sine function from the simulation.	129
5.9	Diagram of sinusoidal network circuit.	130
5.10	Resistor networks and current sources in the sinusoidal spatial filter circuit, (a) the full networks, (b) a single branch.	131
5.11	Simulation of the sinusoidal network circuit.	132
5.12	Circuit model of a p-type floating gate MOSFET.	134
5.13	Illustration of the programmable current element (PCE).	139
5.14	Current bias for the floating-gate pFET, (a) simple current mirror, (b) cascode current mirror	140

5.15	Programmed current at equilibrium state from transient sweep simulation.	141
5.16	Schematic of the PCE.	142
5.17	Transient sweep simulation of programming PCE, ×: the channel current in <i>M3</i> after programming is done, ○: the channel current in <i>M3</i> after switching to filtering mode for PCE without <i>M4a</i> , *: the channel current in <i>M3</i> after switching to filtering mode for PCE with <i>M4a</i> . The output current in <i>M7</i> is very close to the channel current in <i>M3</i> in the filtering mode.	143
5.18	Ratio of the output current to the target current after programming in the filtering mode from transient sweep simulation of PCE.	143
5.19	Drain voltage (left) and floating gate voltage (right) from transient sweep simulation of PCE. ×: when programming is finished, ○: after switching to filtering mode for PCE without <i>M4a</i> , *: after switching to filtering mode for PCE with <i>M4a</i>	144
5.20	Schematic of the bias circuit for the PCE.	145
5.21	The layout of the PCE.	145
5.22	Architecture of the motion image sensor.	146
5.23	Illustration of the PCM.	147
5.24	PCM in programming mode.	150
5.25	PCM in sensing and filtering mode.	152
5.26	Test setup for the PCM.	153
5.27	Programming of 10 nA to an array of PCEs.	154
5.28	Calibration of the output current mirror.	155
5.29	Program a sinusoidal filter.	156
5.30	Output current to target current ratio after programming.	157
5.31	Filtering a constant current input with the <i>sin</i> filter.	158
5.32	Output current to input current ratio as a function of the input current.	158
5.33	Photo of the flyeye test setup.	159

5.34	Mismatch compensation of the EMD velocity tuning, 1 velocity unit = $28.9^\circ/s$	160
5.35	Averaging velocity tuning curve of EMDs for different bias current of low-pass filter, 1 velocity unit = $28.9^\circ/s$	161
5.36	Distribution of output current from all PCEs from four filters.	162
5.37	Mismatch compensation of the EMD response in one filter, 1 velocity unit = $28.9^\circ/s$	163
5.38	Total EMD responses after mismatch compensation from four filters, 1 velocity unit = $28.9^\circ/s$	164
5.39	The sensor and the PCB board.	165
5.40	The integrated sensor and vehicle.	166
5.41	Block diagram of the system.	166
5.42	View angle calibration.	167
5.43	Program four filters: $a_0 = 2$, $a_1 = 2 \cos(\cdot)$, $a_2 = 2 + \cos(2\cdot)$, $b_1 = 2 + \sin(\cdot)$. \circ : programmed current from filter taking minus part of the input, \times : programmed current from filter taking plus part of the input. Dotted line: scaled ideal filter current.	169
5.44	Mean filter outputs at constant angular velocities, positive angular velocity: counter clockwise, negative angular velocity: clockwise.	169
5.45	Three tunnel settings for closed loop experiments.	172
5.46	Trajectories of the vehicle navigating through the tunnel for 20 runs with initial offset $y = -24.1$ cm, dashed line: mean trajectory, $\delta_y = 6.6$ cm.	173
5.47	Trajectories of the vehicle navigating through the tunnel for 20 runs with initial offset $\theta = 21^\circ$, dashed line: mean trajectory, $\delta_y = 9.6$ cm.	174
5.48	Trajectories of the vehicle navigating through the tunnel for 20 runs with no initial offset, dashed line: mean trajectory, $\delta_y = 6.8$ cm.	174
A.1	The transconductance amplifier for the GmC filter with large time constant, (a) schematic, (b) symbol.	183
A.2	The circuit block diagram of the adaptive large time constant filter.	184

A.3	The configuration of the adaptive large time constant filter, (a) programming, (b) low-pass filter, (c) high-pass filter.	186
A.4	Transient simulation of the programming of the adaptive filter.	187
A.5	Sweep AC analysis of the adaptive filter configured as a LPF with the capacitance from 100 fF to 1 pF in the increment of 100 fF.	188
A.6	Sweep AC analysis of the adaptive filter configured as a HPF with the capacitance from 100 fF to 1 pF in the increment of 100 fF.	189
B.1	Previous adaptive floating gate pixel structure.	191
B.2	New adaptive floating gate pixel structure.	191
B.3	Small signal model of the previous pixel structure.	192
B.4	Small signal model of the new pixel structure.	193

List of Abbreviations

κ	subthreshold slope factor
F	Faraday's constant
R	Gas constant
T	Absolute temperature
V_T	Thermal voltage
A/D	Analog to Digital
AC	Alternating Current
BIST	Build-In-Self-Test
CI	Contralateral Inhibited (CI)
CMOS	Complementary Metal-Oxide-Semiconductor
D-FF	D-type Flip Flop
DAQ	Data Acquisition
DC	Direct Current
DIP	Dual In-line Package
DOF	Degree of Freedom
DSP	Digital Signal Processing
DUT	Device Under Test
EMD	Elementary Motion Detector
FD	Figure Detection
FG	Floating Gate
FPGA	Field Programmable Gate Array
FPN	Fixed Pattern Noise
GA	Genetic Algorithm
GND	electrical GrouND
GPIO	General Purpose Interface Bus
HPF	High-Pass Filter
IC	Integrated Circuit
LED	Light Emitting Diode
LMC	Large Monopolar Cell
LPF	Low-Pass Filter
LPTC	Lobula Plate Tangential Cell
LTD	Long-Term Depression
LTP	Long-Term Potentiation
MOSFET	Metal-Oxide-Semiconductor Field Effect Transistor
ND	Null Direction
nFET	n-type Field Effect Transistor
NMDA	N-Methyl-D-Aspartate

PCB	Printed Circuit Board
PCE	Programmable Current Element
PCM	Programmable Current Matrix
PD	Preferred Direction
PFGA	Programmable Floating-Gate Array
PPD	Paired-Pulse Depression
PPF	Paired-Pulse Facilitation
PRNG	Pseudo Random Number Generator
PSC	Pulse Shaping Circuit
PSP	PostSynaptic Potential
PTP	Post-Tetanic Potentiation
pFET	p-type Field Effect Transistor
PSD	Power Spectral Density
REMD	Reichardt Elementary Motion Detector
RMS	Root Mean Square
RNG	Random Number Generator
SDE	Stochastic Differential Equation
SNR	Signal to Noise Ratio
STD	Short-Term Depression
STDP	Spike Timing Dependent Plasticity
UV	Ultra-Violet
VLSI	Very Large Scale Integration
WFI	Wide Field Integration
XOR	Exclusive-OR

Chapter 1

Preamble

1.1 Introduction

The dissertation represents my efforts towards solving engineering challenges using biological computational principles. The elegance of the biological systems is reflected from the underlying biochemical machinery at very low level, to the signal representations, the algorithms for specific tasks, and systems integration and organization at very high level. Specifically, I investigated neural systems at two very different levels, the synaptic level and the sensorimotor integration level, which inspired the similar implementations in VLSI. The first silicon stochastic synapse with short-term depression ever reported in the literature is demonstrated by modulating the influence of noise on the circuit. The depressing silicon stochastic synapse exhibits the behavior similar to its biological counterpart and matching the theory and simulation. An integrated motion image sensor with on-chip optic flow estimation, adaptation, and programmable spatial filtering is also demonstrated. The sensor emulates the fly's eyes to extract motion information from detailed structures of the optic flow field. The sensor is integrated with a ground robot and the robot is able to navigate through simple tunnel environments using the sensor output, in the similar way as the fly's visually-guided navigation. For these two very different VLSI systems, I demonstrate that adaptation is essential for their proper

function and how floating gate MOSFETs in standard CMOS processes can be used to implement adaptation and programmability.

The advancement of semiconductor technology brings us both the opportunities and the challenges. As feature sizes of integrated circuits continue to shrink and demand for low power operation continues to increase, power supply voltages may eventually be reduced to the level where noise becomes significant compared with signals. The signal corruption affects not only analog circuits but digital circuits as well. Specifically, this would cause digital logic circuits to exhibit non-deterministic behavior. This poses serious challenges to the conventional paradigm of deterministic computation based on digital logic circuits. Can we turn the ubiquitous noise into our friend? New computational paradigms that account for this randomness and take advantage of it may achieve robust performance despite the noise [3–8].

In biology, synapses are among the primary locations in neural systems where information is processed and transmitted. Synaptic transmission is a stochastic process by nature, i.e., a presynaptic spike triggers a synaptic vesicle to release its chemical transmitter with a certain probability. The probability determines both the synaptic efficacy and the postsynaptic potential. Synaptic efficacy can increase or decrease within milliseconds after the onset of specific temporal patterns of activity, phenomena known as synaptic facilitation and depression. Recent evidence suggests that short-term synaptic plasticity performs temporal filtering [9, 10] and is involved in many functions such as gain control [11], phase shift [12], coincidence detection, and network reconfiguration [13]. The computational power of dynamic stochastic synapses has been demonstrated in theory [14], and through network simulation [15].

It has been suggested that randomness in ion channel and synaptic transmission increases the energy efficiency of the information coding and processing [16,17]. It has also been shown that depressing stochastic synapses can increase information transmission efficiency by filtering out redundancy in presynaptic spike trains [18,19]. Long-term plasticity that depends on the precise timing of spikes, referred to as spike timing dependent plasticity (STDP) [20,21], has also been observed and modeled as a modulation of release probability based on the pre- and postsynaptic spike timing [22].

Can we create the controllable randomness and implement the similar dynamics for information processing in silicon? We propose a stochastic synapse in silicon where the noise is used in a differential mode to generate randomness. The circuit exhibits stochastic transmission of input spikes, and the transmission probability can be adjusted by input voltage bias, therefore providing the mechanism to further implement activity dependent dynamics. We choose a subtractive single release model for the short-term depression to implement in silicon. The experimental result demonstrates that the depressing silicon stochastic synapse shows similar behavior as the biological synapse.

To obtain stochastic behavior from the silicon implementation, there is another obstacle to overcome, the fabrication mismatch. Fabrication mismatch is the device parameter variation from wafer to wafer, die to die, or even within the same die due to the processing and masking limitation [23–26]. It is a major concern for high performance analog circuits, affecting offset voltages, errors in current mirrors, errors in bias currents, etc. For the stochastic synapse, mismatch would bias the circuit into

the deterministic operation, either transmitting every input spike or none of them. Although it is possible to manually tune the circuit to the stochastic operation, it is apparently a laborious process. Floating-gate MOSFET is a nonvolatile storage unit where charges can be stored at the floating gate for a long period through tunnelling and injection [27, 28]. The charges can be used as a local weight for computation or to cancel the offset from mismatch. It has been applied in many applications such as autozeroing amplifier [29], focal plane online nonuniformity correction [30], adaptive floating-gate comparator [31], fixed pattern noise reduction [32], single transistor synapse [33–35], analog memory [36], current trimming [37], matrix operation [38, 39], and learning in silicon [40, 41]. In our stochastic synapse, we use floating-gate pFET injection configured in negative feedback to automatically program the circuit into the stochastic region. This makes it possible to implement a large array of stochastic synapses on-chip. The floating-gate pFET also provides the capability to implement long-term plasticity in the stochastic synapse.

To fully understand the stochastic operation of the circuit and the effect of noise, we propose to model the circuit using stochastic differential equations (SDEs) and use the numerical solution of the SDEs as the time domain transient analysis. Normally noise effects are analyzed in the frequency domain in terms of power spectral density (PSD). For a linear system, the output noise PSD is related to the input noise PSD by its transfer function, i.e., total output noise PSD is the summation of all noise sources scaled by the power transfer functions of all intervening subsystems [42, 43]. Input referred noise is computed by dividing the total output noise by the system power transfer function for direct comparison with input sig-

nals, i.e. to compute signal noise ratio (SNR). These capabilities are provided by popular circuit simulators. Such frequency domain analysis, however, is not suitable for many circuits, particularly where large signal behavior or nonlinearity is an important characteristic, or where performance depends on transient sample paths and ensemble statistics as for the stochastic synapse. The transient analysis shows how the signal evolves in the presence of noise fluctuations, and how those fluctuations influence the statistics of the signal samples at specific times. Heuristic methods have been proposed before for noise transient analysis by synthesizing time domain noise signals using a sum of sinusoidal signals [44] or a series of pulses [45]. These methods cannot show the interaction of noise with the circuit dynamics. We develop a small signal stochastic model of the circuit. This model gives us a close look at how device parameters and operation biases affect the circuit, and fully elucidates the origin of its stochastic behavior.

We apply the same method to one of the fundamental digital circuits, a CMOS inverter, at low power supply voltages. Understanding the stochastic behavior of digital circuits could potentially help us to find non-conventional computation scheme that makes use of noise in digital circuits. We use numerical simulations of SDEs to obtain time domain transient analysis of the CMOS inverter and magnitude statistics from multiple sample paths. Furthermore, we derive the output distribution at steady state from first principles.

We then turn our attention to another problem for low power, low load, and highly integrated sensor and actuator systems for micro air vehicles. A fly clearly presents us an excellent solution we can learn from. Flies have survived over 300

million years and represent one of the most successful animal groups on our planet (i.e. one tenth of the known species is a fly). Their sensorimotor system seems much simpler than those of larger animals such mammals, with eyes of only 3000 pixels in spatial resolution and a set of 17 muscles controlling the wings [46]. But the system is highly specialized and perfected to tasks directly related to their survival and reproduction. The fly's photoreceptors can operate in a large dynamic range, from single photons to light up to $\sim 10^6$ effectively absorbed photons per second due to the adaptation mechanism [47]. The parallel computation in sensory system to extract behaviorally relevant information and close interaction between sensory and motor systems enable the fly to respond fast. Flies can chase mates at turning velocities of more than $3000^\circ \text{ s}^{-1}$ with delay times less than 30 ms [48] and maintain flight stability within turbulent conditions [49]. They can handle the noisy and uncertain environment well, which is always a challenging problem for engineered systems. The amazing performance is all achieved in a small package with a neural system of less than a million neurons and only 0.1 milligram in weight [50]. In comparison, conventional computer vision based navigation systems rely on CCD imagers and digital microprocessors. Static images are captured along time and sampled in space, and task relevant information is extracted from the huge amount of data in images by the series operation of digital microprocessors. The system is bulky and demands large memory, high power consumption, and long operation time. On the Sojourner rover of the Mars mission, the CCD imagers consumed 5% of the total power and the CPU system consumed 24%, much of which was devoted to the processing of the images [51]. Therefore the fly inspired visually

guided navigation system offer highly efficient solutions to low power, light weight, high speed, and robust sensor and actuator systems for micro air vehicles.

Flying insects have immobile eyes with a fixed focal length. With poor spatial resolution and lack of binocular stereopsis, but better temporal resolution, flying insects extract most visual cues from image motion for visually guided behavior. When an insect flies around, moving patterns of illuminance generate local image shifts on the retina, called optic flow. The optic flow contains information about self-motion, object motion, and spatial layout of the environment, playing critical roles in vision based navigation such as flight stabilization, centering response, object avoidance, landing, predator attack, chasing behavior, etc. [52–56]. Optic inputs go through several layers of processing and project to the lobula plate, where lobula plate tangential cells (LPTCs) have large dendritic trees, integrating visual information across a large field. There are about 60 different LPTCs on each side identified by their characteristic anatomy and response [48]. Recent experimental results revealed that there is fine spatial structure of the motion direction sensitivity in the receptive field of LPTCs beyond their preferred global direction [57]. The structure matches typical optic flow patterns induced by self-motion. It has been postulated that such matched filter structures correspond to neuronal models of the external world [58]. LPTCs may compute specific self-motion parameters by filtering its vast visual input stream through synaptic connections.

The matched filter type of operation of optic flow has been formulated in a feedback control scheme, as the wide field integration (WFI) based navigation [2, 59–61]. It was shown that motion cues can be extracted via circular Fourier decomposition

of the optic flow and used in a feedback control loop to drive the vehicle towards a desired state or trajectory. It was also demonstrated that the feedback methodology is sufficient to explain some of the experimentally observed honeybee flight behavior [60]. To have a coherent solution of the motion cue estimation inspired from the fly, we evaluate the fly inspired optic flow estimator, the elementary motion detector (EMD) [62–65], in this framework. Simulation suggests that EMD is a possible candidate of optic flow estimation for the WFI based navigation although noisy output, nonlinear response, and contrast dependence could cause problems. Since flying insects have limited view angles, which are much less than 2π , as used in circular Fourier decomposition, we investigate motion cues recovery and feedback control from a limited view angle and extend previous result to the general case of limited view angles. This also helps to alleviate technical difficulties associated with constructing a full circular motion sensor.

We then develop a single chip solution to the WFI based navigation, aiming to generate the motion cues on-chip and feed them directly to the vehicle for navigation control. The chip has optic flow estimation and programmable spatial filters. Previously EMD has been used for navigation where EMD responses across the field of view were spatially averaged to provide the velocity information about the motion [66–69]. And it is the only parameter extracted and used. In this case, the individual differences among EMDs caused by mismatch is not critical. While for the WFI based navigation, detailed spatial structure of the optic flow field is used to extract multiple dynamic and kinematic parameters, therefore the intrinsic difference among EMDs caused by mismatch would distort the original optic

flow spatial pattern and pose a serious problem. Again, we need to overcome mismatch as we do for the stochastic synapse. We use the floating-gate pFET injection configured in negative feedback, but in a different way. It implements not only the mismatch compensation, but also the programming of the filters. The same floating-gate pFET also functions as the multiplier for filtering. Therefore we use a very compact structure and simple mechanism to achieve three functions at the same location. The method can also be applied to other sensory front-end in integrated sensors where on-chip spatial processing is required and distortion from fabrication mismatch among sensor units has to be reduced. Although the sensor is geared for micro air vehicles, we first demonstrate the concept using a prototype with the sensor integrated with a ground robot. Both open loop and closed loop system test are performed.

Insect neural systems are relatively simple, and it is feasible to emulate many of their structures and functions using current technology. Not only does this enable biologically-inspired robots with high performance and efficiency, it also allows us to test our understanding of the biological systems [70, 71].

As demonstrated, floating-gate MOSFET is a power technique to bring adaptation into the circuits. In the appendices, I further explore its application in two other proposed circuits: adaptive large time constant filter and adaptive floating-gate pixel.

1.2 Organization

The thesis is organized as follows.

In chapter 2, I first introduce the basic biology of the synapse and synaptic plasticity, and explain the stochastic nature of the synapse and its connections to plasticity. I briefly review previous efforts in developing adaptive VLSI synapse for VLSI learning systems, followed by the discuss of the difficulty of introducing randomness to VLSI learning system and how randomness can be generated and used under the context of random number generator (RNG) integrated circuit (IC). I then introduce the silicon stochastic synapse we proposed and the method to incorporate either the short-term or the long-term plasticity. I describe the implementation of the short-term depression based a subtractive single release model. Statistical characteristics and spike transmission dynamics are investigated using the fabricated chip. The experimental results are presented that demonstrate the good randomness and robustness of the silicon stochastic synapse, similar behavior from the short-term depression as its biological counterpart, and good agreement with theory and simulation.

In chapter 3, I first briefly review circuit noise. Then I introduce the mathematic concepts and tools for stochastic modeling and simulation: from the definition of random variable and process, to stochastic integral and SDE, to numerical solution of the SDE. I illustrate the basic technique to write the SDE for a voltage node and extend it to the silicon stochastic synapse. Using transient analysis based on the numerical simulation of the SDEs, I demonstrate the operation mechanism

of the silicon stochastic synapse. Furthermore, I introduce a small signal stochastic model, which elucidates the dynamics behind the stochastic behavior and the influences from device parameters and operation biases. Similar treatment is given to a CMOS inverter to understand the stochastic behavior of digital elements when operating at very low voltage supplies. Analytical solution of steady state output distribution is obtained and compared with numerical simulations.

In chapter 4, I first introduce the fly's vision system, the sensorimotor system, and the visually guided navigation. I then introduce the optic flow WFI based navigation and EMD. Using a simulink model, I evaluate the EMD and its variations in the feedback control of WFI based navigation. I extend the theoretic result to the general case of limited view angles.

In chapter 5, I describe the single chip solution to the WFI based navigation. The motion image sensor is consist of three major functional modules: EMD array, programmable current matrix (PCM), and control logic. The details are given for each circuit components. Two approaches to spatial filters, resistive networks and nonvolatile storage, are illustrated and compared. In the final design, nonvolatile storage approach is used. A novel structure, programmable current element (PCE), is proposed to achieve mismatch compensation, filter programming, and filtering operation at the same location. The post-sensory computational core is formed by PCM, a matrix of PCEs. The architecture and the operation of the motion image sensor are described, followed by the experimental results of PCE programming and EMD mismatch compensation. At the end, I describe the effort for integration of the sensor with a ground robot. Experimental results are shown and discussed.

In chapter 6, I summary the thesis work and discuss open problems and future directions of the research.

In appendix A, I introduce the large time constant GmC filter using indirect feedback and the problem from fabrication mismatch. I describe a solution to compensate the mismatch and how the circuit can be configured for adaptation and filtering operation. Transient simulation of the adaptation and AC analysis for the frequency response are shown.

In appendix B, I propose an improved adaptive floating-gate pixel. I introduce both circuits and compare their frequency responses.

1.3 Contributions

The following is a summary of contributions from my Ph.D. research.

1.3.1 Dynamic VLSI Stochastic Synapse

- Propose and fully characterize the first VLSI stochastic synapse circuit with high quality of randomness.
- Propose methods to incorporate both short-term and long-term activity dependent plasticity in the VLSI stochastic synapse.
- Propose and simulate a circuit model of short-term depression.
- Design a compact circuit of short-term depression in the VLSI stochastic synapse.

- Demonstrate the short-term depressing behavior that is similar to its biological counterpart and matches the theoretical and simulation results.

1.3.2 Stochastic Circuit Modeling

- Propose a method for stochastic circuit modeling using SDEs and transient analysis of circuit behavior based on numerical solution of SDEs.
- Propose a stochastic model of the VLSI stochastic synapse and perform the transient analysis.
- Propose a small signal stochastic model of the stochastic synapse relating device parameters, operating biases, noise, and circuit dynamics.
- Analyze the stochastic behavior of a fundamental digital circuit, a CMOS inverter, at low supply voltages.

1.3.3 Fly Inspired Integrated Approach to Autonomous Navigation

- Evaluate the EMD as the optic flow estimator in WFI based navigation.
- Generalize previous theoretical result of WFI based navigation to limited view angles and demonstrate the result in simulation.
- Propose a circuit to achieve mismatch compensation, filter programming, and filter operation all in one place.
- Design and characterize a low power integrated motion image sensor with on-chip optic flow estimation, adaptation, and programmable spatial filters.

- Integrate the sensor with a ground robot and demonstrate autonomous navigation in simple tunnel environments. This is the first single VLSI sensor solution to EMD navigation based on WFI.

Chapter 2

Silicon Stochastic Synapse with Plasticity

2.1 Introduction

2.1.1 Neuron and Synapse

Nervous system is essential for animals to interact with the outside world. It is responsible for information communication and computation. Nervous system is a massive interconnected network, where neurons are the basic structure units, and synapses form the connections. In the human nervous system there are about 10^{12} neurons and 10^{15} synapse in brain [72].

A typical neuron has three parts: 1) a tree of processes called dendrites, which receive signals from other neurons; 2) cell body, where signals from dendrites are integrated and outputs are generated; 3) a long process called axon, along which neural signals from the cell body travel a distance to the end, and pass to other neurons through synapses.

Neural signals propagate by membrane potentials: graded potentials or action potentials. The membrane potential is determined by the ion concentration inside and outside of the neuron and the ion channel permeability, where Na^+ , K^+ , and Cl^- ions are the three major players. The permeability determines the conductance of the channel. The membrane potential is given by the Goldman-Hodgkin-Katz

(GHK) equation [72]:

$$V = \frac{RT}{F} \ln \frac{P_K [K^+]_{out} + P_{Na} [Na^+]_{out} + P_{Cl} [Cl^-]_{in}}{P_K [K^+]_{in} + P_{Na} [Na^+]_{in} + P_{Cl} [Cl^-]_{out}} \quad (2.1)$$

where R is the gas constant, F is Faraday's constant, and T is absolute temperature. At room temperature 293 K, $\frac{RT}{F} \approx 58mV$. $[\cdot]$ are the ion concentrations, and P are the permeability of the ion channels. In general $[K^+]_{in} > [K^+]_{out}$, $[Na^+]_{in} < [Na^+]_{out}$, and $[Cl^-]_{in} < [Cl^-]_{out}$. They are maintained by the active transport of ions. When one ion channel is dominant, the membrane potential is mainly determined by that ionic reversal potential $E_s = \frac{RT}{Fz} \ln \frac{[s]_{out}}{[s]_{in}}$, where z is the valence of the ion.

Action potentials, also known as spikes or neural impulses, are used by neurons to transmit neural signals over long distance. They are all-or-none transient signals that are mediated by voltage and time dependent ion channel conductance. The mechanism is best understood through Hodgkin and Huxley's study of squid giant axon. A simple count of the action potential is as follows. When the neuron is at rest, the membrane is most permeable to K^+ , so the rest potential is close to E_K . Stimulus causes depolarization of the membrane. Once the membrane is above the threshold voltage, the conductance of the Na^+ increases bringing more inward current to the neuron, therefore depolarizing the membrane further. This is a fast regenerative process that drives the membrane potential to E_{Na} eventually. At the same time, the conductance of K^+ increases slowly causing more outward current, and the conductance of Na^+ decreases after it reaches its maximum. Eventually, the conductance of K^+ is much higher than Na^+ so that the membrane is polarized

back to its resting potential. In the end, all ion channel conductances recover back to their resting state. Na^+ - K^+ pump is responsible to restore the intracellular and extracellular concentration of Na^+ and K^+ [72].

When spikes reach the end of the axons, the presynaptic terminals, they activate the voltage dependent Ca^{2+} channels and Ca^{2+} flows in. This causes the vesicles containing neurotransmitters to fuse with the presynaptic membrane, thus to release the neurotransmitters to the synaptic cleft. The neurotransmitters diffuse to the postsynaptic membrane and bind to the receptors. This triggers the opening of the ion channels directly or through other messenger paths. The opening of the ion channel causes either positive or negative postsynaptic potential (PSP). Neurotransmitters are recycled back to presynaptic terminals to reform vesicles. PSPs at the dendritic terminals propagate to the cell body and are integrated to determine whether a postsynaptic spike will be generated [72].

It has been observed that the PSP is the unit multiples of the PSP from the spontaneous release of a single vesicle, and the fluctuation of the PSP is hypothesized due to the variation of the number of vesicles released, known as quantum hypothesis. It has also been observed that at central synapses transmission proceeds in an all-or-none fashion. This leads to the hypothesis that at most one vesicle can be released per spike per active zone, and the release is a stochastic event. The exact mechanism is not clear, and could be due to lateral inhibition across presynaptic membrane, constraint of single releasable vesicle per active zone, or postsynaptic

receptor saturation [73]. The synaptic weight has been modeled as [74]:

$$R = npq \tag{2.2}$$

where n is the number of quantal release sites, p is the probability of release per site, and q is some measure of the postsynaptic effect.

2.1.2 Synaptic Plasticity

Synapses are among the primary locations in neural systems where information is processed and transmitted. They undergo constant changes in order to learn from and adapt to the ever-changing outside world. Synaptic efficacy can increase or decrease within milliseconds after the onset of specific temporal patterns of activity and can last from milliseconds to days. The variety of synaptic plasticities differ in the triggering condition, time span, and involvement of pre- and postsynaptic activity.

In a paired-pulse facilitation (PPF), the amplitude of response from the second input pulse increases. After a train of input pulses, the response increases as well, the early phase is called augmentation, and the later phase is called post-tetanic potentiation (PTP). It is suggested that all these enhancements are due to an increased release probability, thus a presynaptic phenomenon. A simplified explanation is that the residual Ca^{2+} from previous pulse or pulse train increases the release probability. Depression can also occur after a single pulse or a pulse train, known as paired-pulse depression (PPD) or depletion. It is attributed to the decrease of release probability or number of release site [72, 74].

The long term form of enhancement or depression after a train of input stimuli are known as long-term potentiation (LTP) and long-term depression (LTD). They are associated with learning and memory. Both presynaptic and postsynaptic activities are involved, and the rising of intracellular Ca^{2+} in the postsynaptic neuron plays an important role. Donald Hebb proposed the famous rule governing the long-term plasticity: in the Hebbian case, the synapse efficacy increases when there is a causal relation between the presynaptic and postsynaptic firing, in the anti-Hebbian case, the synapse efficacy decreases when the postsynaptic firing occurs without presynaptic firing or vice versa. A close investigation of the relation between the synaptic plasticity and the precise firing time discovered the so called spike timing dependent plasticity (STDP) [20, 21]. If a presynaptic spike precedes a postsynaptic spike within a critical time window, the synaptic efficacy increases. Conversely, if a presynaptic spike arrives right after a postsynaptic spike, the synaptic efficacy decreases. Table 2.1 summarizes the different forms of synaptic plasticity [74].

The regulation of the vesicle release probability has been considered as the underlying mechanism for various synaptic plasticities. Many models are built to match the experimental data. Senn, Markram, and Tsodyks [22] proposed an algorithm for modifying neurotransmitter release probability based on pre- and postsynaptic spike timing. The release probability is upregulated if the presynaptic spike occurs up to 50 ms before the postsynaptic spike and downregulated if the presynaptic spike occurs up to 50 ms after the postsynaptic spike. The up- and downregulation are mediated by the N-methyl-D-aspartate (NMDA) receptors located at the postsynap-

Table 2.1: Synaptic plasticity

Phenomenon		Duration	Location	
Short Term	Enhancement	Paired-pulse facilitation	100 ms	pre
		Augmentation	10 s	pre
		Post-tetanic potentiation	1 min	pre
	Depression	Paired-pulse depression	100 ms	pre
Depletion		10 s	pre	
Long Term	Long-term potentiation		> 30 min	pre/post
	Long-term depression		> 30 min	pre/post

tic membrane. Tsodyks and Markram [75] modeled the synaptic transmission using the dynamics among three states (effective, inactive, recovered) of the connection resource, and showed that the neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. The rate of synaptic depression is determined by the release probability and dictates whether firing rate or temporal coherence of presynaptic spikes is signaled to the postsynaptic neurons.

2.1.3 Adaptive VLSI Synapse

The remarkable performance of biological neural computation has inspired research in VLSI neural systems where efforts are made to build neurally-inspired learning systems in silicon and use them as powerful experimental tools to study neural systems as well. Similar to biological neural systems, silicon neuron is the functional unit of the system and silicon synapses connect the neurons to form the

network. Adaptive VLSI synapses have been extensively studied and developed as the central units for adaptation and learning. Floating-gate single transistor synapses have been proposed where adaptation can be achieved locally in parallel and over long times [33–35]. Häfliger and Mahowald [76] developed a synapse with weight change depending on the temporal correlation of spikes. Silicon synapses with short-term depression (STD) have been developed and modeled, with the weight of the synapse implemented by a gate voltage which decreases after each presynaptic spike and recovers between the spikes [77–79]. Short-term facilitation and depression have also been implemented using a current mirror integrator [80]. Synapses with temporally-asymmetric Hebbian learning rules have been implemented [81] as well as synapses with STDP [82].

All these adaptive synapses are deterministic VLSI synapses. Alternatively stochastic synapses transmit spikes according to a transmission probability. Stochastic synapses have been difficult to implement in VLSI because it is hard to properly harness the probabilistic behavior, normally provided by noise. Although stochastic behavior in integrated circuits has been investigated in the context of random number generators (RNGs) [83], these circuits either are too complicated to use for a stochastic synapse or suffer from poor randomness. Therefore other approaches were explored to bring randomness into the systems. Stochastic transmission was implemented in software using a lookup table and a pseudo random number generator [84]. Stochastic transition between potentiation and depression has been demonstrated in bistable synapses driven by stochastic spiking behavior at the network level for stochastic learning [85, 86].

2.1.4 Randomness in VLSI

Stochastic behavior in integrated circuits has been investigated in the context of RNGs, which have broad application in cryptography, scientific computing, and stochastic computing. There is growing interest in integrated circuit (IC) RNGs which can be easily integrated in systems-on-chip, for information protection, built-in-self-test (BIST), or hardware implementation of genetic algorithms (GA). Random numbers are generated on-the-fly inside the system without feeding the sequence externally. Normally random bits are generated which can be assembled into integers or fractions. Random numbers with uniform distribution are generated and could be transformed to random numbers with other distributions by inverse cumulative density functions [87].

The quality of randomness of the RNGs directly influences the system performance. Pseudo-RNG (PRNG) generates sequences using a deterministic algorithm, such as congruential generators and shift-register generators [88], so the sequence inevitably repeats and becomes predictable. For some applications this repeated pattern will cause performance degradation. For other applications the predictability of the random sequence has serious consequences, i.e., it jeopardizes the secrecy of the cryptography. Many hardware RNGs are PRNGs, which implement hardware friendly RNG algorithms in VLSI or FPGA. These hardware PRNGs suffer similar problems to software PRNGs.

A true RNG is nondeterministic and unpredictable, often relying on the randomness from physical phenomena, such as radioactive decay, photon arrival, ther-

mal noise from a resistor, or shot noise from a Zener diode. For IC true RNG, there are three commonly used techniques in the literature. The direct amplification technique amplifies small AC voltages produced by a noise source and generates digital signals by thresholding with a clocked comparator [89]. This method is sensitive to interference from other noise sources and requires significant shielding. The oscillator sampling method [90] uses a slow clock to sample a much faster clock in order to acquire a digital signal. Randomness comes from the phase jitter of the slow clock, which is not always adequate to provide randomness of good quality. The speed of the RNG is limited by the slow clock as well. Electronic chaotic systems have also been used to generate unpredictable signals [91–93]. Petrie and Connelly [83] created behavioral models for these three different IC-compatible methods for producing random numbers, and combined them together to build a robust true RNG immune to non-random influences. Other techniques are also developed for RNG [94–96]. These circuits either are too complicated to use for a stochastic synapse or suffer from poor randomness.

In the following section we present the first silicon stochastic synapse. The circuit is compact (~ 15 transistors) and the experimental results demonstrated good randomness as well as the feature of probability tuning. It can also be used as a true RNG IC [97]. Furthermore, we propose the method to implement stochastic synapse with plasticity and demonstrate the implementation of short-term depression (STD) by modulating the *probability* of spike transmission. Like its deterministic counterpart, this probabilistic synapse provides filtering on individual spike train inputs; its stochastic character, however, creates the possibility of a broader range of compu-

tational primitives such as rate normalization of Poisson spike trains, probabilistic multiplication, or coincidence detection.

2.2 Silicon Stochastic Synapse

The stochastic synapse uses competition between two intrinsic circuit noise sources to generate random events. The core of the structure is a clocked, cross-coupled differential pair comparator (Fig. 2.1(a)) with input voltages V_{i+} and V_{i-} . The same circuit has previously been used in an adaptive comparator for offset cancellation [31]. When V_{clk} is logic high, $V_{o+} \approx V_{o-}$. When V_{clk} becomes logic low, transistor M5 shuts off, V_{o+} and V_{o-} are nearly equal and the circuit is in its metastable state. If V_{g+} is significantly higher than V_{g-} , initially V_{o+} increases and V_{o-} decreases. So the channel current of M3 increases and the channel current of M4 decreases, which further drives V_{o+} up and V_{o-} down. This is a positive feedback procedure that drives V_{o+} close to V_c and V_{o-} close to ground. If V_{g+} is significantly lower than V_{g-} , the opposite happens. When V_{g+} is very close in value to V_{g-} , circuit noise that produce fluctuations in the drain currents of M1 and M2 will dictate the outcome. The final result depends on the sign of the imbalance, $V_{o+} - V_{o-}$, which triggers positive feedback after transistor M5 shuts off. The detailed model of the circuit will be given in chapter 3.

Fabrication mismatch in an uncompensated circuit would likely permanently bias the circuit to one solution. In this circuit, floating gate inputs to a pFET differential pair allow the mismatch to be compensated. Since there is no direct electrical

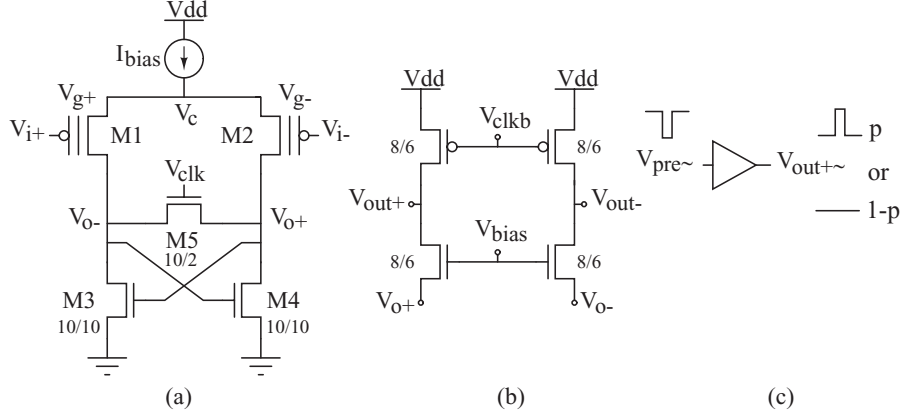


Figure 2.1: Stochastic synapse circuit: (a) clocked cross-coupled differential pair comparator, (b) dynamic output buffer, (c) input-output behavior. $\frac{W_{1,2}}{L_{1,2}} = \frac{14}{2}, \frac{4}{12}$ have been used.

connection to the floating gate, its potential is determined by capacitive coupling to nearby nodes and charge stored on the node. The voltage at the floating gate can be modified with fine resolution by hot-electron injection or tunnelling mechanisms. The circuit uses a nominal power supply of 5 V during normal operation, a 5.5 V supply for injection during mismatch compensation, and a 15 V supply for tunnelling used only during initialization. By controlling the common-mode voltage of the floating gates, we operate the circuit such that hot-electron injection occurs only on the side where the output voltage is close to ground. Injection current is a function of the drain current, gate-drain voltage V_{gd} , and source-drain voltage V_{sd} . The higher the V_{gd} and V_{sd} are, the higher the injection current is. When one floating gate is higher in voltage than the other, the comparator output on that side will be pulled low. With proper common input voltage, this will provide enough V_{sd} and V_{gd} for injection to happen only on the side pulled low, thus reduce the floating gate

voltage. Over multiple clock cycles hot-electron injection works in negative feedback to bring close the floating gate voltages, therefore bring the circuit into stochastic operation. When the procedure is left long enough, it equilibrates the floating gate voltages and reaches the equilibrium point at which the circuit outputs around 50% probability of the two alternative results. We use a dynamic buffer (Fig. 2.1(b)) driven by the complement of the same clock, to convert the voltages at V_{o+} and V_{o-} into rail-to-rail pulse signals.

When the circuit functions as a stochastic synapse, the inverted presynaptic spike train $V_{pre\sim}$ is used as V_{clk} , $V_{out+\sim}$, the inverted V_{out+} , is used as the transmitted spike train. When a presynaptic spike comes, $V_{out+\sim}$ either goes high (with probability p) or stays low (with probability $1 - p$), emulating the stochastic transmission of spikes, as shown in Fig. 2.1(c). It can be used with deterministic synapse of spiking neurons where the stochastic spike output $V_{out+\sim}$ drives the deterministic synapse to generate postsynaptic potentials or currents. Good randomness and robustness against interference have been demonstrated [97]. The compactness of the circuit, good randomness, and insignificant coupling between adjacent circuits make it possible to implement multiple stochastic synapses on-chip for VLSI learning systems.

The transmission probability can be adjusted by changing the input offset or the floating gate charges. The higher V_{g+} is, the lower p is. The probability tuning function is closely fitted by an error function $f(v) = 0.5 \left(1 + \text{erf} \left(\frac{v-\mu}{\sqrt{2}\delta} \right) \right)$, where μ is the input offset voltage for $p = 50\%$, δ is the standard deviation characterizing the spread of the probability tuning, and $v = V_{i-} - V_{i+}$ is the input offset volt-

age [97]. Synaptic plasticity can be implemented by dynamically modulating the probability. Input offset modulation is suitable for short-term plasticity. Fig. 2.2 shows the block diagrams of silicon stochastic synapse with short-term plasticity. Short-term depression is triggered by the transmitted input spikes V_{tran} to emulate the probability decrease because of depletion. V_{dw} controls the magnitude of depression. Short-term facilitation is triggered by the input spikes V_{pre} to emulate the probability increase because of the Ca^{2+} accumulation. V_{fw} controls the magnitude of facilitation. Other controls are not shown in the figure such as the time constant control. Nonvolatile storage at the floating gate is suitable for long-term plasticity. Fig. 2.3 shows the block diagram of silicon stochastic synapse with long-term plasticity. STDP can be implemented by modulating the probability depending on the precise timing relation between the presynaptic spike V_{pre} and postsynaptic spike V_{post} . Hebbian learning is triggered when V_{pre} precedes V_{post} in a time window determined by $V_{\tau pre}$ and results in the increase of the release probability, therefore the increase of the synapse efficacy. Whereas anti-Hebbian learning is triggered when V_{post} precedes V_{pre} in a time window determined by $V_{\tau post}$ and results in the decrease of the release probability, therefore the decrease of the synapse efficacy. V_{hw} and V_{ahw} control the magnitude of the Hebbian learning and anti-Hebbian learning respectively.

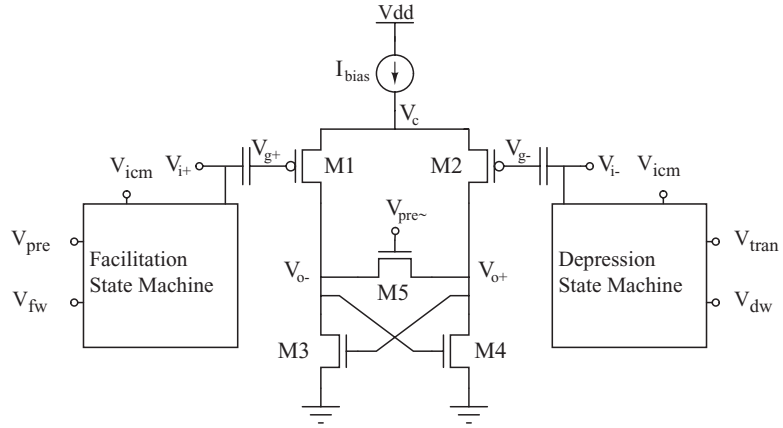


Figure 2.2: Block diagram of silicon stochastic synapse with short-term plasticity.

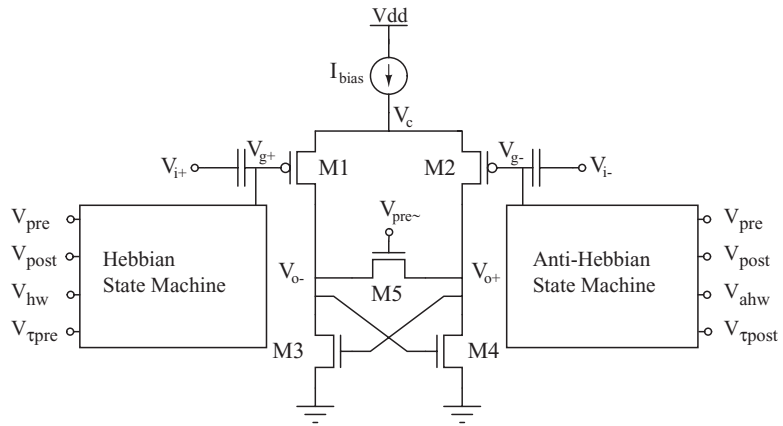


Figure 2.3: Block diagram of silicon stochastic synapse with long-term plasticity.

2.3 Stochastic Synapse with Short-Term Depression

Although long-term plasticity has attracted much attention because of its apparent association with learning and memory, the functional role of short-term plasticity has only recently begun to be understood. Recent evidence suggests that short-term synaptic plasticity performs temporal filtering [9, 10] and is involved in many functions such as gain control [11], phase shift [12], coincidence detection, and network reconfiguration [13]. From the perspective of information transmission, it has also been shown that depressing stochastic synapses can increase information transmission efficiency by filtering out redundancy in presynaptic spike trains [18]. Activity dependent short-term changes in synaptic efficacy at the macroscopic level are determined by activity dependent changes in vesicle release probability at the microscopic level. We will focus on short-term depression (STD) here.

2.3.1 Short-term Depression Circuit Model

STD during repetitive stimulation results from a decrease in released vesicles. Since there is a finite pool of vesicles, and released vesicles cannot be replenished immediately, a successful release triggered by one spike potentially reduces the probability of release triggered by the next spike. We propose an STD model based on our VLSI stochastic synapse that closely emulates the simple subtractive single release model [18, 98, 99]. A presynaptic spike that is transmitted reduces the input offset voltage v at the VLSI stochastic synapse by Δv , so that the transmission probability $p(t)$ is reduced. Between successful releases, v relaxes back to its maxi-

mum value v_{max} exponentially with a time constant τ_d so that $p(t)$ relaxes back to its maximum value p_{max} as well. The model can be written as

$$v(t_+) = v(t_-) - \Delta v, \text{ successful transmission at } t \quad (2.3)$$

$$\tau_d \frac{dv(t)}{dt} = v_{max} - v(t) \quad (2.4)$$

$$p(t) = f(v(t)) \quad (2.5)$$

For an input spike train with Poisson arrivals, the model can be expressed as a stochastic differential equation

$$dv = \frac{v_{max} - v}{\tau_d} dt - \Delta v \cdot dN_{p \cdot r(t)} \quad (2.6)$$

where $dN_{p \cdot r(t)}$ is a Poisson counting process with rate $p \cdot r(t)$, and $r(t)$ is the input spike rate. By taking the expectation $E(\cdot)$ on both sides, we obtain a differential equation

$$\frac{dE(v)}{dt} = \frac{v_{max} - E(v)}{\tau_d} - \Delta v \cdot E(p)r(t) \quad (2.7)$$

When v is reduced, the probability that it will be reduced again becomes smaller. v is effectively constrained to a small range where we can approximate the function $f(v) = 0.5 \left(1 + \text{erf} \left(\frac{v - \mu}{\sqrt{2}\delta} \right) \right)$ by a linear function $f(v) = av + 0.5$, where $\mu = 0$ for simplicity. We can then solve for $E(p)$ at steady state:

$$p_{ss} \approx \frac{av_{max} + 0.5}{1 + a\Delta v\tau_d r} \approx \frac{p_{max}}{a\Delta v\tau_d r} \approx \frac{p_{max}}{\Delta p\tau_d r} \propto \frac{1}{r} \quad (2.8)$$

Therefore the steady state mean probability is inversely proportional to the input spike rate when $a\Delta v\tau_d r \gg 1$. This is consistent with the original simple subtractive single release model [18, 100] and STD model at the macroscopic level [11].

2.3.2 Simulation

We simulated the STD circuit model (2.3)-(2.5). We use the function $f(v) = 0.5 \left(1 + \operatorname{erf} \left(\frac{v}{\sqrt{2 \cdot 2.16}} \right) \right)$, obtained from the best fit of the experimental data. Initially v is set to 5 mV which sets p_{max} close to 1. Although the transformation from v to p is nonlinear, both simulation and experimental data show that this implementation exhibits behavior similar to the model with the linear approximation and the biological data. Fig. 2.4 and 2.5 show that the mean probability is a linear function of the inverse of the input spike rate at various Δv and τ_d for high input spike rates. Both Δv and τ_d affect the slope of the linear relation at the similar trend suggested by (2.8): the bigger the Δv or the bigger the τ_d , the smaller the slope is. Fig. 2.6 shows a simulation of the transient probability for a period of 200 ms.

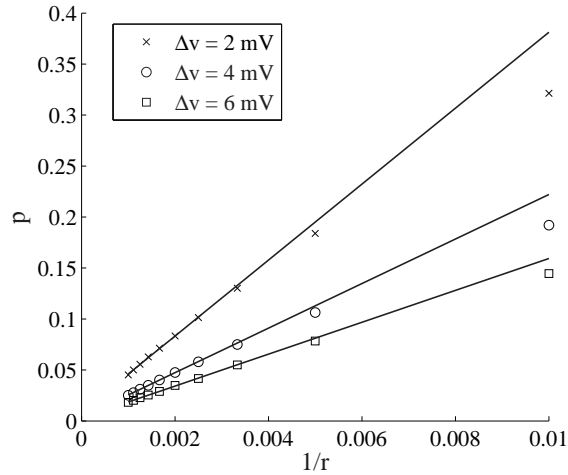


Figure 2.4: Mean probability as a function of input spike rate for $\Delta v = 2, 4, 6$ mV. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The solid lines show the least mean square fit for input rates from 400 Hz to 1000 Hz. $\tau_d = 100$ ms.

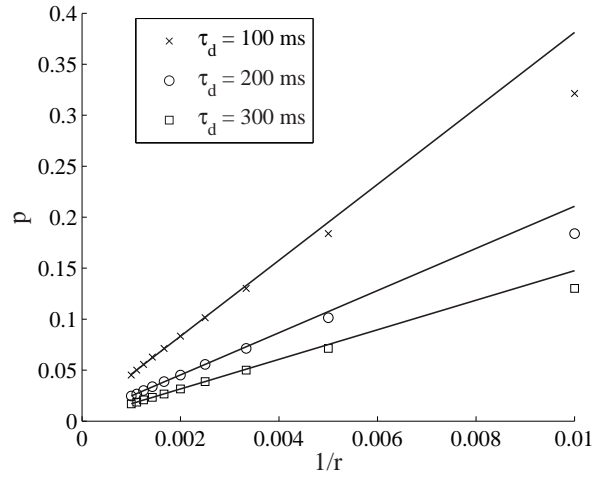


Figure 2.5: Mean probability as a function of input spike rate for $\tau_d = 100, 200, 300$ ms. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The solid lines show the least mean square fit for input rates from 400 Hz to 1000 Hz. $\Delta v = 2$ mV.

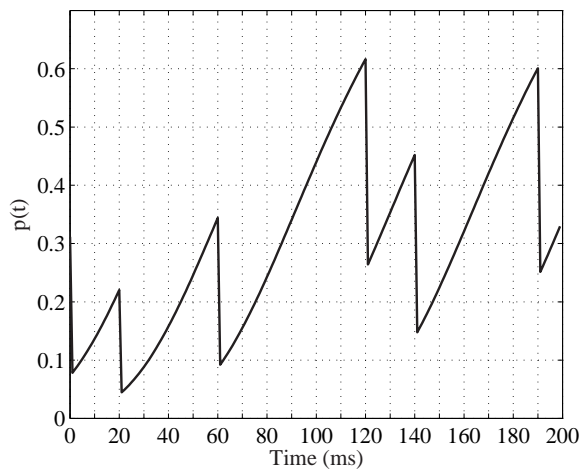


Figure 2.6: Simulated probability trajectory over 200 ms period. $r = 100$ Hz, $\tau_d = 100$ ms, $\Delta v = 2$ mV.

Fig. 2.7 shows the autocorrelation of the output spike trains for different time constants τ_d . The value at 0 is not plotted in order to show the negative autocorrelation better. The negative autocorrelation is the direct result from the STD. The time interval of the negative autocorrelation is apparently dependent on the time constant. The longer the time constant is, the longer the period of the negative autocorrelation is. When the time constant is small enough, p can recover to its maximum value before the next spike arrival, therefore the transmission won't be affected by previous transmissions. Fig. 2.8 shows the PSD of the output spike trains from the same results shown in Fig. 2.7. The power reduction at low frequencies is suggested to reduce information redundancy in the input spike trains. The larger the time constant is, the lower the frequencies where power reduction reside. Fig. 2.9 shows the autocorrelation of the output spike trains for different voltage drops Δv . It suggests that Δv mainly affects the magnitude of the negative autocorrelation rather than the time scale.

2.3.3 Circuit Implementation

We implemented this model using our stochastic synapse circuit. Fig. 2.10 shows the block diagram of the circuit. Both inputs are restored up to an equilibrium value V_{icm} by tunable resistors. To change the transmission probability we only need to modulate one side of the input, in this case V_{i-} . The resistor and capacitor provide for exponential recovery of the voltage to its equilibrium value. The input V_{i-} is modulated by transistors M6 and M7 based on the result of the previous spike

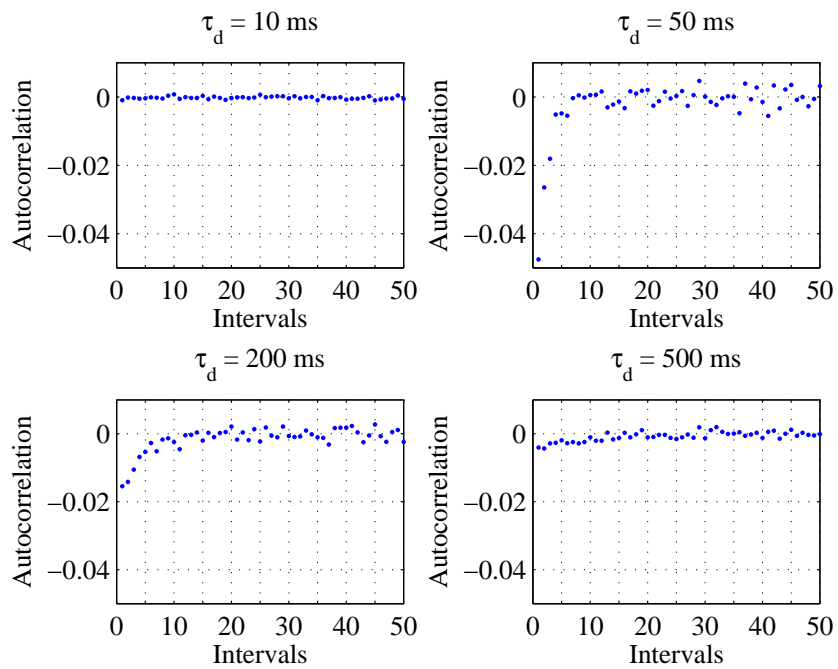


Figure 2.7: Autocorrelation of output spike trains for different time constants τ_d , each spike interval is 10 ms.

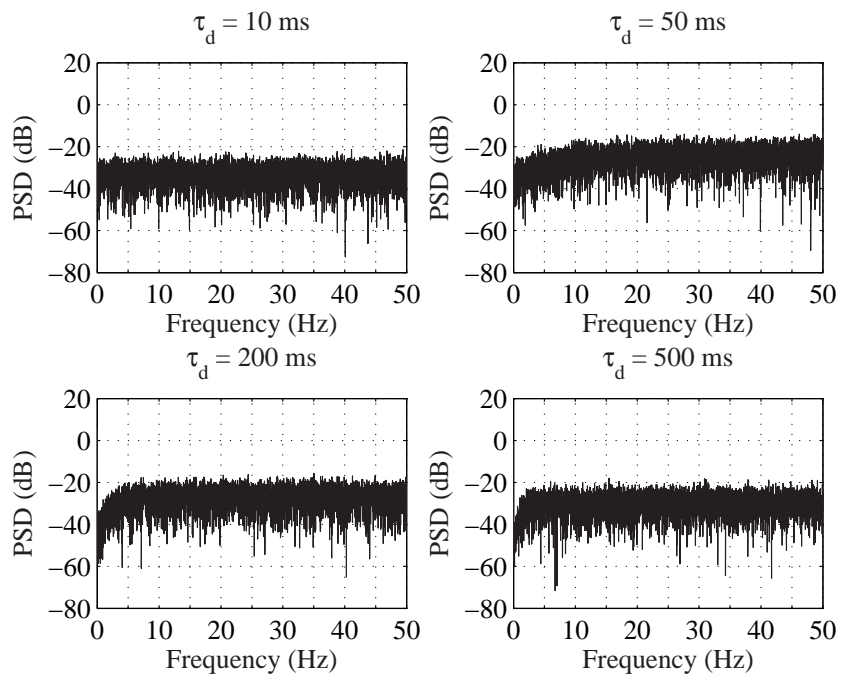


Figure 2.8: Power spectral density of output spike trains for different time constants

τ_d .

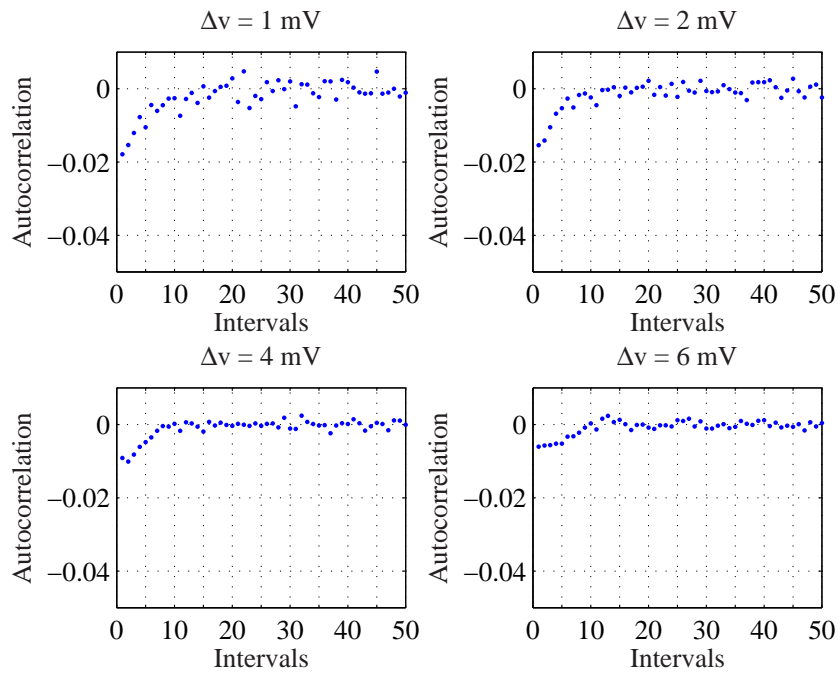


Figure 2.9: Autocorrelation of output spike trains for different voltage drop Δv , each spike interval is 10 ms.

transmission. Every time a spike is transmitted successfully, a pulse with height V_h and width T_p is generated at V_p . This pulse discharges the capacitor with a small current determined by the V_w and reduces V_{i-} by a small amount, thus decreasing the transmission probability. The tunable resistors are implemented by subthreshold pFETs operating in the ohmic region. Their resistance is controlled by the gate voltage of the pFETs, V_r . When V_{i-} is reduced, the probability that it will be reduced again becomes smaller. Since the probability tuning only occurs in a small voltage range (~ 10 mV), the change of V_{i-} is limited to this small range as well. Under this special condition, the resistance implemented by the subthreshold pFET is linear and large ($\sim G\Omega$). With capacitance as small as 100 fF, the exponential time constant is tens of milliseconds and adjustable. Similar control circuits can be applied to V_{i+} to implement short-term facilitation. The update mechanism would then be driven by the presynaptic spike rather than the successfully transmitted spike.

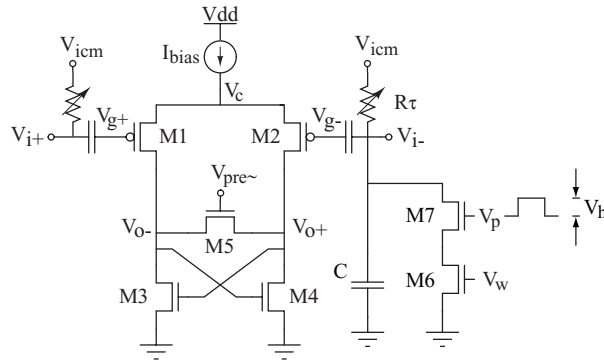


Figure 2.10: Block diagram of silicon stochastic synapse with STD.

In the first implementation, a pulse shaping circuit (PSC) is designed to generate pulses with adjustable width and height at V_p . Fig. 2.11 shows the block

diagram of the PSC. Part (a) generates a rising edge triggered pulse from input spike, and V_t controls the pulse width T_p . When a spike comes in, D flip-flop (D-FF, transmission gate based) is triggered on its rising edge and its Q_b goes low. It opens the pFET $M1$ to charge the capacitor C_m . The output of the Schmitt trigger V_o changes from logic high to low quickly once its input V_i increases above the threshold voltage. This clears the D-FF and Q_b becomes high, thus shuts off $M1$. It also turns on nFET $M3$ so that a small current determined by V_t discharges C_m . Once V_i is below a certain threshold, V_o goes high again, and this shuts off $M3$. The pulse generating procedure is then finished. The mechanism guarantees that one input spike generates only one output pulse. The intrinsic hysteresis of the Schmitt trigger provides enough dynamic range for pulse width adjustment. Part (b) is to reduce the height of the pulse to V_h , therefore to reduce the capacitance coupling effect when it is used to control the input at the stochastic synapse.

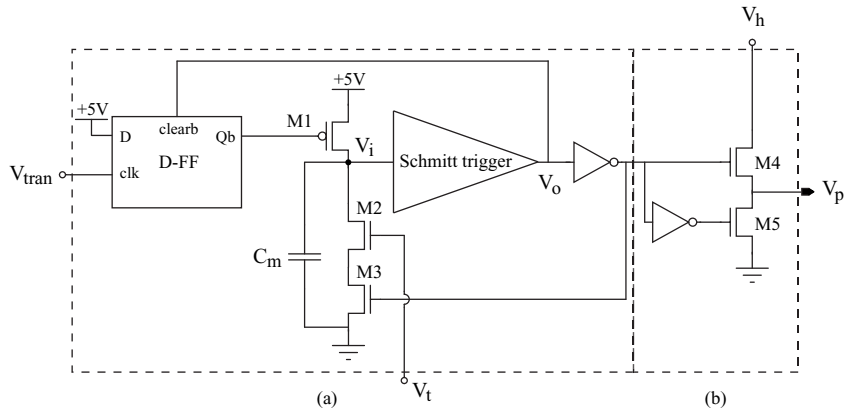


Figure 2.11: Block diagram of pulse shaping circuit.

Fig. 2.12 shows one transient simulation of V_{i-} driven by 5 transmitted spikes at 50 Hz. At each spike, V_{i-} drops by about 2.14 mV. Between spikes V_{i-} increases

back to its equilibrium value exponentially. At $V_r = 2$ V and $V_{icm} = 2.5$ V, the recovery time constant is 51.2 ms. The spikes in the trace are caused by coupling from the pulses due to parasitic capacitances in the circuit.

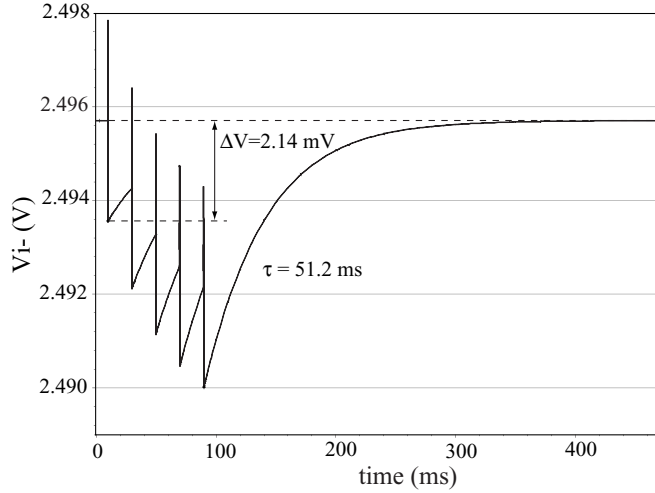


Figure 2.12: Transient simulation of V_{i-} driven by 5 transmitted spikes at 50 Hz, $V_t = 0.7$ V, $V_w = 0.4$ V, $V_h = 0.5$ V.

Although this PSC provides much flexibility to control the amount of the voltage drop at V_{i-} , it is complicated compared with the rest of the circuit. In the later implementation, part (a) is dropped and only part (b) is used. This results in a much compact implementation of the stochastic synapse with STD, as shown in Fig. 2.13. The stochastic synapse part of the circuit is shown in the dashed box. V_{tran} is the transmitted presynaptic spike V_{pre} . The pulse width of V_{tran} will be the same as the input spike V_{pre} and cannot be adjusted separately. The pulse height can still be adjusted by V_h . The experimental results to be shown later demonstrate that V_w itself is adequate to adjust the voltage drop at V_{i-} . This makes the circuit much more compact for integration. The layout size of the STD is $35 \mu\text{m} \times 32.2$

μm and the layout size of the stochastic synapse is $151.9 \mu\text{m} \times 91.7 \mu\text{m}$. A 2-to-1 multiplexer with size $35 \mu\text{m} \times 30 \mu\text{m}$ is used to enable or disable the STD. The extra components on the left provide for future implementation of short-term facilitation and also symmetrize the stochastic synapse, improving its randomness.

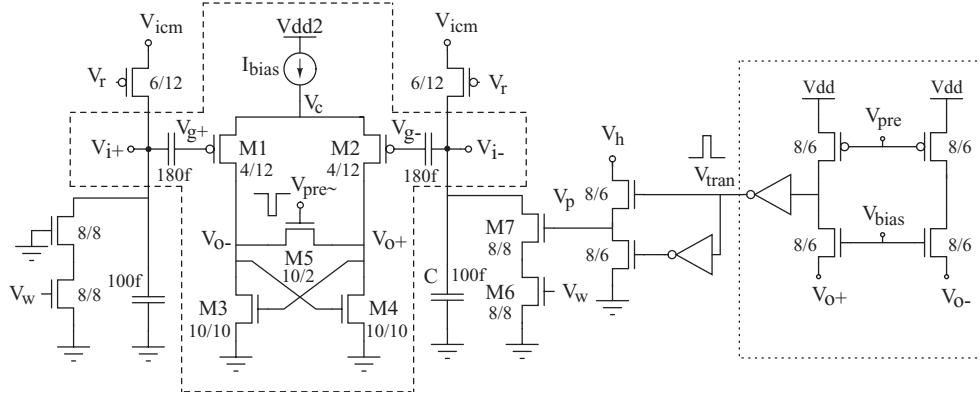


Figure 2.13: Schematic of a compact implementation of stochastic synapse with STD.

2.4 Experimental Results

The circuits have been fabricated in a commercially-available $0.5 \mu\text{m}$ CMOS process with 2 polysilicon layers and 3 metal layers. It uses a nominal power supply of 5 V for normal operation. The differential pair comparator uses a separate power supply for hot-electron injection. Each floating-gate pFET has a tunnelling structure, which is a source-drain connected pFET with its gate connected to the floating node. A separate power supply provides the tunnelling voltage to the shorted source and drain (tunnelling node). When the tunnelling voltage is high enough (14-15V), electron tunnels through the silicon dioxide, from floating gate to the tunnelling

node. We use this phenomenon to remove electrons from the floating gate. Alternatively Ultra-Violet (UV) activated conductances may be used to remove electrons from the gate to avoid the need for special power supplies.

To begin the test, we first remove residual charges on the floating gates in the stochastic synapse. Although we can manually adjust the input offset to bias the circuit in the stochastic region, the negative feedback operation of hot-electron injection described above can automatically bias the circuit for stochastic operation. We raise the power supply of the differential pair comparator to above 5.5 V to enable hot-electron injection. During the procedure, we observe the output spike $V_{out+\sim}$. Initially there is no spike output at all, or there is a spike for every input spike, indicating a deterministic operating point set by the offset from mismatch. Eventually random spike output is observed, indicating that the circuit is operating stochastically. We can halt the injection by lowering the power supply to 5 V. During this procedure, STD is disabled, so that the probability at this operating point is the synaptic transmission probability without any dynamics.

The injection mechanism should be engaged for time long enough to reach the desired steady state probability. If the injection is engaged for a long time, the probability approaches 50%, and the common voltage of V_{g+} and V_{g-} continues to drop until injection ceases. We can also monitor the probability and disable the injection at a specific probability value. Since the injection can only move the probability towards 50%, to set a probability above 50%, an initial offset must be provided to set the probability to 100%, while to set a probability below 50%, an initial offset must be provided to set the probability to 0.

2.4.1 Stochastic Synapse Tested as RNG

We first test the quality of the randomness of the circuit. Just as functioning as the stochastic synapse, the circuits can be viewed as a RNG as well. At each clock low, a random bit is generated at $V_{out+\sim}$, either V_{dd} or ground. We tested the circuit with the clock as fast as 200 kHz, limited by our data acquisition system. Up to 200 kHz, statistical characteristics are consistent. The power consumption of the circuit is about 10-44 μ W for bias current from 1 nA to 1 μ A.

2.4.1.1 Statistical Tests

For the first test of randomness and independence, we examined the autocorrelation and cross-correlation of generated bit sequences. The experimental results match the theory closely, i.e., an independent, identically distributed (i.i.d.) random bit sequence $s(k)$ with probability $prob(s(k) = 1) = p$ has an autocorrelation function $A(n) = E(s(k)s(k+n))$, where $A(n) = p$ for $n = 0$, and $A(n) = p^2$ for $n \neq 0$, and its power spectrum density (PSD) $S(f)$ is flat across all frequencies except for a DC component from the nonzero mean, $S(0) = \frac{np^2}{f_s}$, where n is the total number of the samples and f_s is the sampling frequency. The total power is p . Two i.i.d. sequences with probability p have a cross-correlation function $C(n) = s(k)t(k+n) = p^2$ for all n and the cross-spectral density is also flat across all frequencies except for a DC component, where $S(0) = \frac{np^2}{f_s}$. Fig. 2.14 shows the autocorrelation for the bit sequence from one RNG and cross-correlation between bit sequences from two RNGs. Fig. 2.15 shows the PSD for one bit sequence at sampling frequency 1 kHz.

Fig. 2.16 shows the PSD of the cross correlation between two random bit sequences at sampling frequency 1 kHz.

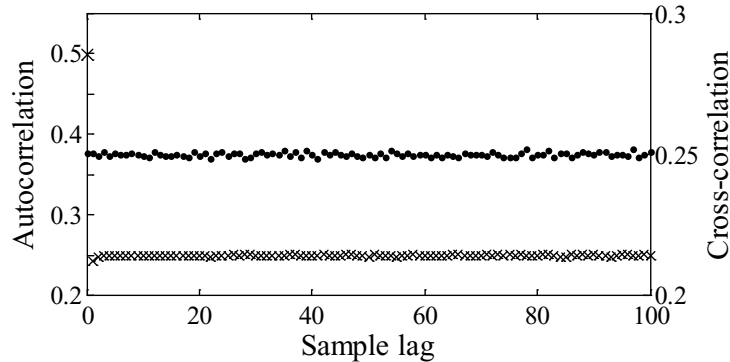


Figure 2.14: Autocorrelation of the bit sequence from one RNG (\times) and cross-correlation between bit sequences from two RNGs (\cdot). For clarity, only data up to a sample lag of 100 is shown.

Existing methods can readily remove small biases in the probability caused by injection mismatch [101]. An exclusive-OR (XOR) of multiple independent random sequences will exponentially converge to an equal probability of 0 and 1 and simultaneously eliminate slight anti-correlation between adjacent bits, caused by kickback noise in the comparator and visible in Fig. 2.14 for $n = 1$. For a rigorous test of the RNG circuit, we applied a battery of benchmark statistical tests developed by the National Institute of Standards and Technology (NIST) [102]. For each statistical test, the suite uses a probability value (P-value) to assess whether a bit sequence passes a statistical test or not. A test statistic is computed from the bit sequence and the P-value is the probability of obtaining a larger value than the obtained. The test statistic is constructed in such a way that a random sequence generates a

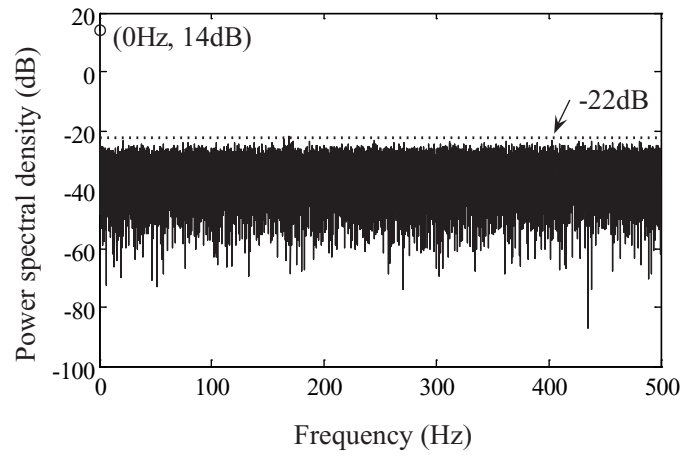


Figure 2.15: Power spectral density of one bit sequence at sampling frequency 1 kHz (similar results are obtained for measurements up to 200 kHz).

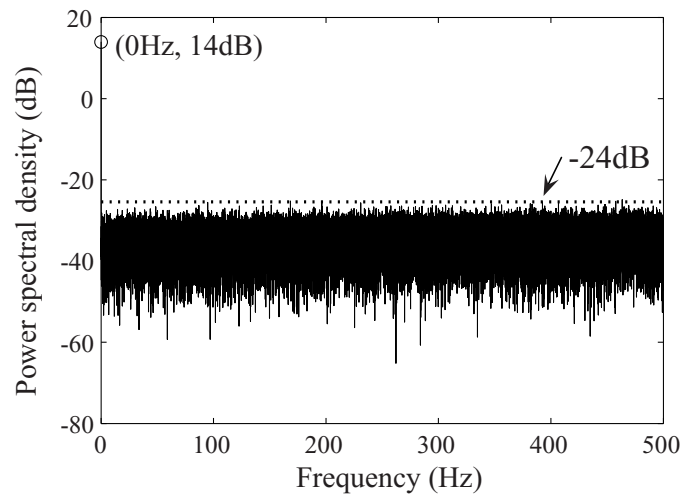


Figure 2.16: Power spectral density of cross correlation between two random bit sequences.

small test statistic. Therefore small P-values indicates non-random sequences. For a given significance level α , the sequence fails the test if its P-value $< \alpha$ [102]. We use $\alpha = 0.01$.

For a sample size of n sequences, the results from the statistical test are further evaluated using either the proportion of test sequences passing the test or the uniformity of the P-values from all sequences. A confidence interval is computed as $(1 - \alpha) \pm 3\sqrt{\frac{(1-\alpha)\alpha}{n}}$. If the proportion of passing sequences is above the lower boundary of the interval, the sequences are considered as passing the test. The uniformity of the P-values is examined by χ test from the histogram of the P-values where a P-value of the P-values is computed. When the P-value is above 0.0001, the P-values from all sequences are uniformly distributed, indicating good randomness of the sequences [102].

20 sequences of 10^6 bits from the XOR of four RNGs were evaluated using the suite. Table 2.2 summarizes the results. For a sample size of 20, with a significance level of 0.01, the minimum required pass rate is 0.923. So the sequences pass all the tests except 3 templates out of 148 templates in the non-overlapping template matching test and the random excursion test. For the random excursion test, 7 out of 20 samples do not satisfy the criterion for the test to proceed and are considered non-random. For the rest of 13 samples, with a significance level of 0.01, the minimum pass rate is 0.907, thus they pass the test. All the P-values from the statistical tests pass the uniformity test.

Table 2.2: NIST statistical test results

Test	Passing Rate
Frequency	1
Frequency within a block	0.95
Cumulative sum (forward)	1
Cumulative sum (reverse)	1
Runs	1
Longest run of ones in a block	1
Random binary matrix rank	1
Discrete Fourier transform	1
Non-overlapping template matching (148 tests)	1 (119)
	0.95 (26)
	0.90 (3)
Overlapping template matching	0.95
Maurer's universal statistical	1
Approximate entropy	1
Random excursions (8 tests)	1 (7)
	0.9231 (1)
Random excursion variant (18 tests)	1 (18)
Serial (2 tests)	1 (2)
Linear complexity	1

2.4.1.2 Adjustable Probability

The probability of the bit sequence can be adjusted by tuning the DC input voltage applied between V_{i+} and V_{i-} while the circuit is operating near the metastable state. Fig. 2.17 shows the probability as a function of the input offset voltage. At each offset voltage, sequences of 10^5 bits are collected and partitioned into 10 sub-sequences from which the mean and standard deviation of the probability are computed. Input offset can be biased to produce very low probabilities (measured as low as 0.004% in Fig. 2.17) that historically have been difficult to obtain reliably. The function is closely fitted by an error function $f(v) = 0.5 \left(1 + \operatorname{erf} \left(\frac{v-\mu}{\sqrt{2\delta}} \right) \right)$, where $\mu = 0.71$ mV and $\delta = 2.16$ mV.

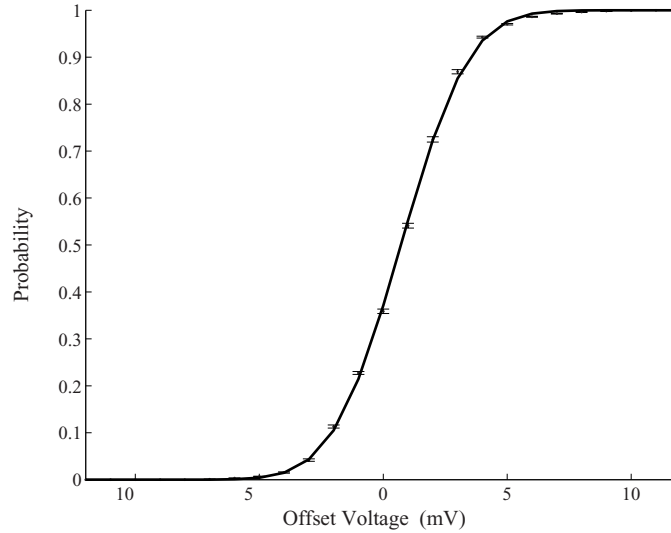


Figure 2.17: Output probability $p(V_{o+} > V_{o-})$ as a function of input offset $v = V_{i+} - V_{i-}$, the solid line is $f(v) = 0.5 \left(1 + \operatorname{erf} \left(\frac{v-\mu}{\sqrt{2\delta}} \right) \right)$, where $\mu = 0.71$ and $\delta = 2.16$.

2.4.1.3 Interference

To test the robustness of the circuits, we evaluated its performance against several common sources of interference such as power supply noise, digital noise, and substrate noise. We use the difference ΔD between the PSD value at DC and the maximum value in the band excluding DC as an indicator of the interference noise power that is coupled into the PSD of the random bit sequence. The measured value without intentionally adding interference is 36dB (Fig. 2.15). We injected sinusoidal signals of different frequencies (10Hz, 100Hz, and 1kHz) and amplitudes (1mV, 5mV, and 10mV) onto the power supply voltage. The lowest ΔD were 33dB, 26dB, and 21dB for amplitudes 1mV, 5mV, and 10mV respectively. We used an on-chip shift register as one example of a digital circuit to evaluate the impact of nearby digital circuitry on the RNGs. The lowest ΔD observed was 27dB. We also injected noise into the substrate by driving 10Hz, 100Hz, and 1kHz square waves (amplitudes up to 2V) through ESD-protected pads. The interference from these square waves was negligible, with the lowest ΔD at 33dB.

The above experimental results demonstrated the good randomness and robustness which make the circuit a very good candidate for stochastic synapse. The unique feature of probability tuning enables us to further incorporate synaptic plasticity.

2.4.2 First Generation Stochastic Synapse with STD

We first show the result from the first generation stochastic synapse with STD. Once the stochastic synapse operates in the stochastic region, we turn on STD. We use a signal generator to generate pulse signals which serve as input spikes. Although spike trains are better modeled by Poisson arrivals, the averaging behavior should be similar for deterministic spike trains which make testing easier. We collect output spikes from the depressing stochastic synapse at an input spike rate of 100 Hz with constant intervals. We divide time into bins according to the input spike rate so that in each bin there is either 1 or 0 output spike. In this way, we convert the output spike train into a bit sequence $s(k)$. We then compute the normalized autocorrelation, defined as $A(n) = E(s(k)s(k+n)) - E^2(s(k))$, where n is the number of time intervals between two bits. $A(0)$ gives the variance of the sequence. For two bits with distance $n > 0$, $A(n) = 0$ if they are independent, indicating good randomness, and $A(n) < 0$ if they are anticorrelated, indicating the depressing effect of preceding spikes on the later spikes. Fig. 2.18 shows the autocorrelation of the output spike train. There is significant negative correlation at small time intervals and little correlation at large time intervals, as expected from STD. Fig. 2.19 shows the PSD of the output spike train. Clearly, the PSD is reduced at low frequencies.

Normally redundant information is represented by positive autocorrelation in the time domain, which is characterized by power at low frequencies. By reducing the low frequency component of the spike train, redundant information is suppressed

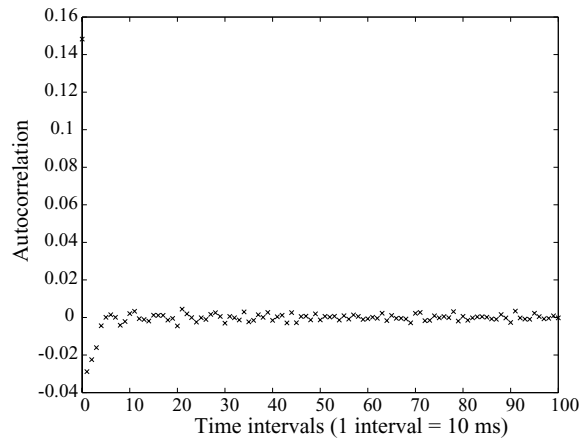


Figure 2.18: Autocorrelation of the output spike train from the stochastic synapse with STD for an input spike rate of 100 Hz. Autocorrelation at time zero represents the sequence variance. Negative autocorrelation at short time intervals indicates STD.

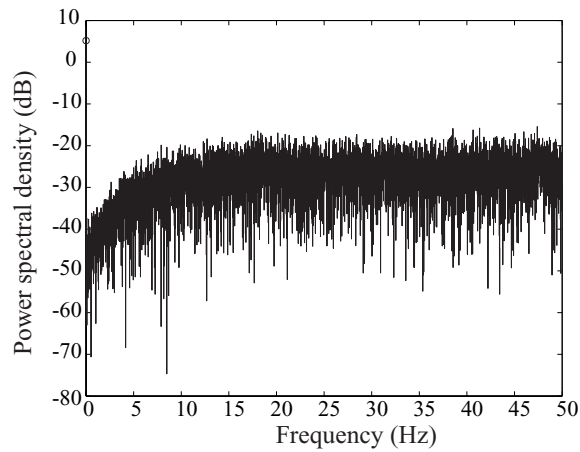


Figure 2.19: Power spectral density of the output spike train from the stochastic synapse with STD for an input spike rate of 100 Hz. Lower PSD at low frequencies indicates STD.

and overall information transmission efficiency is improved. If the negative autocorrelation of the synaptic dynamics matches the positive autocorrelation in the input spike train, the redundancy is cancelled and the output is uncorrelated [18].

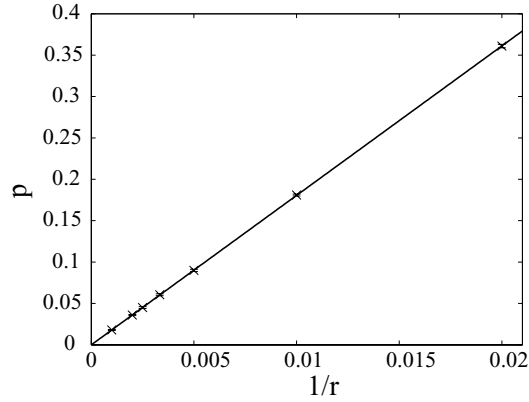


Figure 2.20: Mean probability as a function of the input spike rate. Data were collected at input rates of 50 Hz, 100 Hz, 200 Hz, 300 Hz, 400 Hz, 500 Hz, and 1000 Hz. The solid line shows the least mean square fit $p(x) = 18x - 3.7 \times 10^{-5}$.

We collected output spikes in response to 10^4 input spikes at various input spike rates and plotted the mean probability as a function of the input spike rate. Fig. 2.20 shows that the mean transmission probability is inversely proportional to the input spike rate. This matches the theoretical prediction in (2.8) very well. By scaling the probability with the input spike rate, the synapse tends to normalize the DC component of input frequency and reserve the dynamic range, thus avoiding saturation due to fast firing presynaptic neurons and retaining sensitivity to less frequently firing neurons [11].

Fig. 2.21 shows the mean probability as a function of the input spike rate for different combinations of V_t and V_w . The slope is bigger for larger V_t and smaller

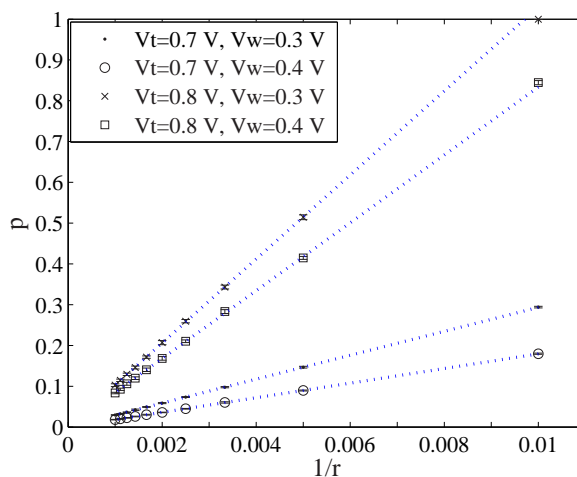


Figure 2.21: Mean probability as a function of the input spike rate for various combinations of V_t and V_w . $V_r = 1.60$ V, and $V_{icm} = 2$ V. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit for input rates from 200 Hz to 1000 Hz.

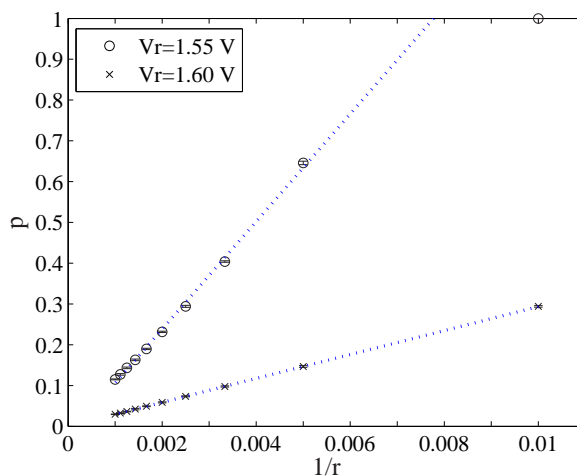


Figure 2.22: Mean probability as a function of the input spike rate for $V_r = 1.55$ and 1.60 V. $V_t = 0.8$ V, $V_w = 0.3$ V, and $V_{icm} = 2$ V. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit for input rates from 200 Hz to 1000 Hz.

V_w . Both V_t and V_w determine the amount of voltage drop Δv at V_{i-} , therefore determine the slope according to (2.8). The larger V_t is, the shorter the pulse is, thus the smaller Δv is. The smaller V_w is, the smaller the current is, thus the smaller Δv is. See the circuit schematics in Fig. 2.10 and Fig. 2.11. Fig. 2.22 shows the mean probability as a function of the input spike rate for $V_r = 1.55$ V and 1.60 V, where $V_t = 0.8$ V and $V_w = 0.3$ V. The slope is bigger for smaller V_r because the equivalent resistance R of the pFET decreases when V_r decreases, thus $\tau_d = RC$ decreases with V_r , and the slope $k \propto \frac{1}{\tau_d}$ increases.

2.4.3 Second Generation Stochastic Synapse with STD

The experiment for the second generation stochastic synapse with STD follows the same procedure as the first one. We use $I_{bias} = 100$ nA. The power consumption by the short-term depression part of the circuit is much smaller than the stochastic synapse part of the circuit. The total power consumption is about 10 μ W.

Fig. 2.23 shows the autocorrelation of the output spike trains at two different V_r for an input spike train of 100 Hz. There is significant negative correlation at small time intervals and little correlation at large time intervals, as expected from STD. Fig. 2.24 shows the PSD of the output spike trains from the same data shown in Fig. 2.23. Clearly, the PSD is reduced at low frequencies. The time constant of STD increases with V_r so that the larger V_r is, the longer the period of the negative autocorrelation is and the lower the frequencies where power is reduced. This agrees with simulation results. Fig. 2.25 shows the simulation result using the estimated

τ_d from $V_r = 1.59$ V. The experimental and simulation results show close similarity.

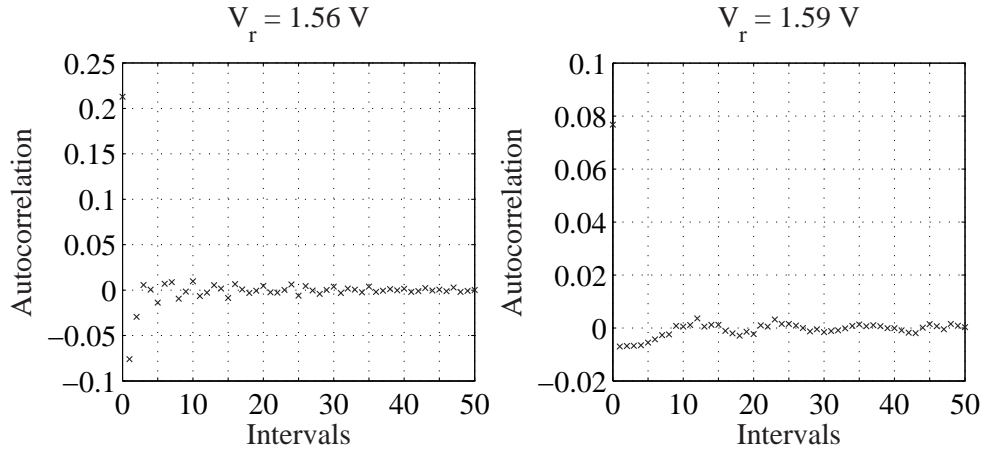


Figure 2.23: Autocorrelation of output spike trains from the silicon stochastic synapse with STD for an input spike rate of 100 Hz. Autocorrelation at time zero represents the sequence variance, and negative autocorrelation at short time intervals indicates STD.

We investigate the relation between mean transmission probability and the input spike rate. Since the pulses that modulate V_{i-} have the same pulse width as input spikes, we can also investigate the effect of pulse width on depressing behavior directly by varying input spike pulse width. We also conduct more detailed study of the effect of various parameters on the depressing behavior. We collect output spikes in response to 10^4 input spikes at input spike rates from 100 Hz to 1000 Hz with 100 Hz intervals. Fig. 2.26 shows that the mean transmission probability is inversely proportional to the input spike rate for various pulse widths when the rate is high enough. This matches the theoretical prediction in (2.8) very well. The slope of mean probability decreases as the pulse width increases. Since the pulse width

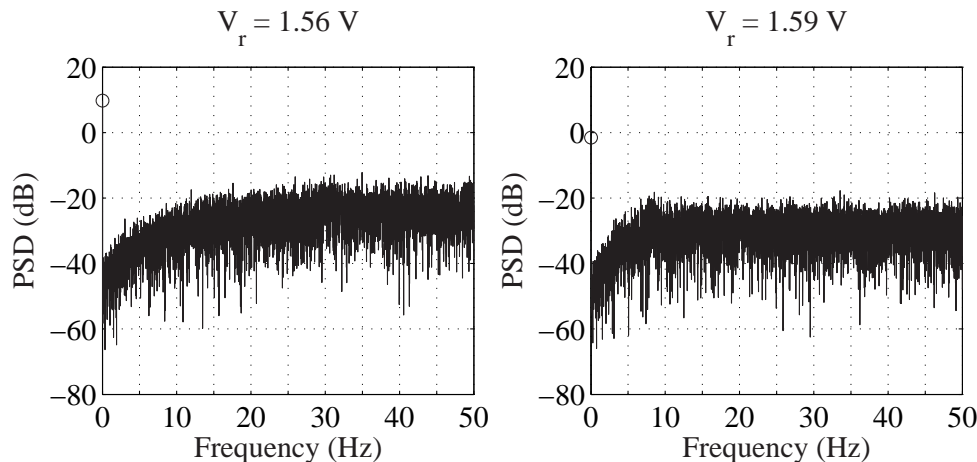
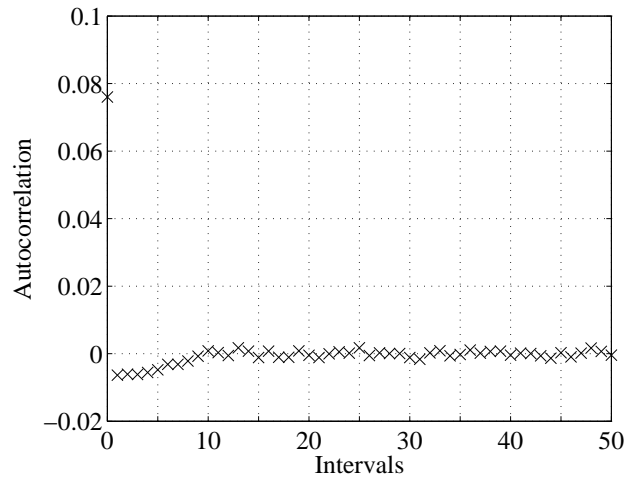


Figure 2.24: Power spectral density of output spike trains from the silicon stochastic synapse with STD for an input spike rate of 100 Hz. Lower PSD at low frequencies indicates STD.

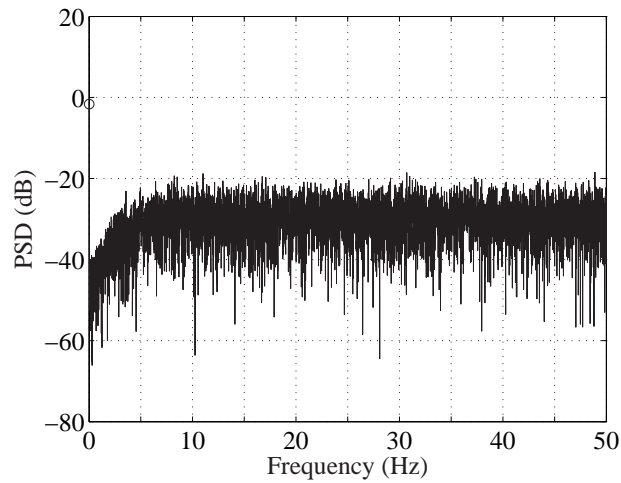
determines the discharging time of the capacitor at V_{i-} , the larger the pulse width, the larger the Δv is and the smaller the slope is. Fig. 2.27 shows that $a\Delta v\tau_d$ scales linearly with the pulse width. The discharging current is approximately constant, thus Δv is proportional to the pulse width.

We perform the same experiments for different V_r and V_w . As V_r increases, the slope of mean transmission probability as a linear function of $\frac{1}{r}$ decreases as shown in Fig. 2.28. This is due to the increasing $\tau_d = RC$, where the equivalent resistance R from the pFET increases with V_r . Fig. 2.29 shows that $a\Delta v\tau_d$ is approximately an exponential function of V_r , indicating that the equivalent R of the pFET is approximately exponential to its gate voltage V_r .

For V_w , the slope of mean transmission probability decreases as V_w increases as shown in Fig. 2.30. This is due to the increasing Δv with V_w . Fig. 2.31 shows



(a) Autocorrelation.



(b) Power spectral density.

Figure 2.25: Characterization of the output spike train from the simulation of the stochastic synapse with STD. $r = 100$ Hz, $\tau_d = 220$ ms, $\Delta v = 5$ mV, $V_{max} = 5$ mV.

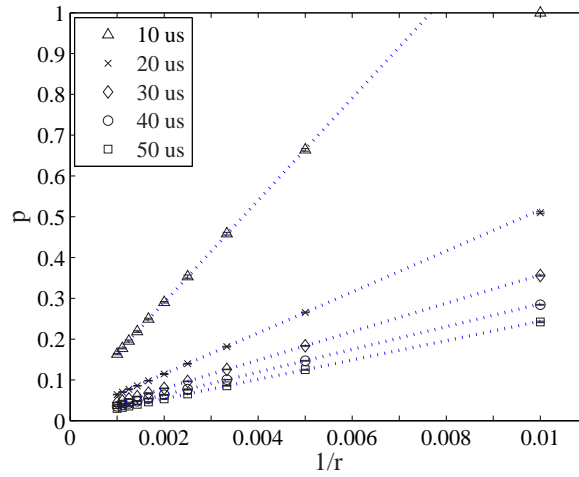


Figure 2.26: Mean probability as a function of input spike rate for pulse width $T_p = 10, 20, 30, 40, 50 \mu s$. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit from 200 Hz to 1000 Hz.

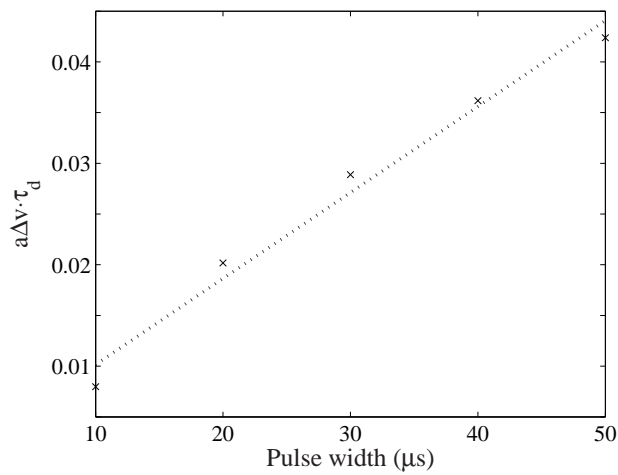


Figure 2.27: $a\Delta v\tau_d$ as a function of the pulse width. The dotted line shows the least mean square fit, $f(x) = 0.0008x + 0.0017$.

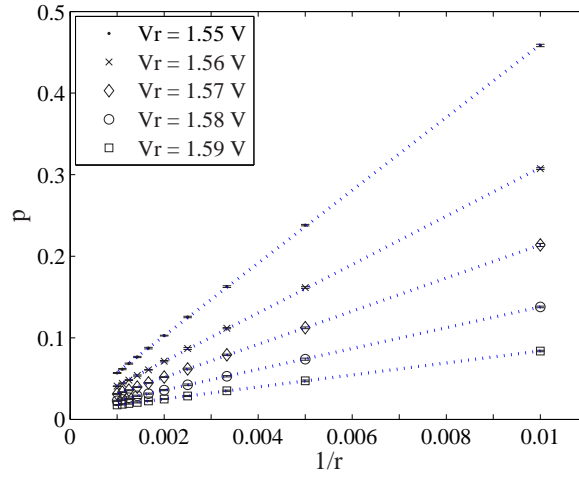


Figure 2.28: Mean probability as a function of the input spike rate for $V_r = 1.55$, 1.56, 1.57, 1.58, 1.59 V. Data were collected at input rates from 100 Hz to 1000 Hz at 100 Hz intervals. The dotted lines show the least mean square fit.

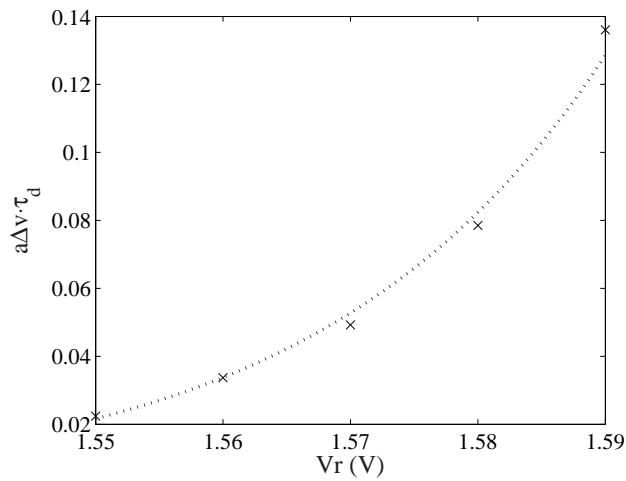


Figure 2.29: $a\Delta v\tau_d$ as a function of V_r . The dotted line shows the least mean square fit, $f(x) = e^{(44.54x-72.87)}$.

that $a\Delta v\tau_d$ is approximately an exponential function of V_w , indicating that the discharging current from the transistor $M6$ is approximately exponential to its gate voltage V_w . This matches the I-V characteristics of the MOSFET in subthreshold.

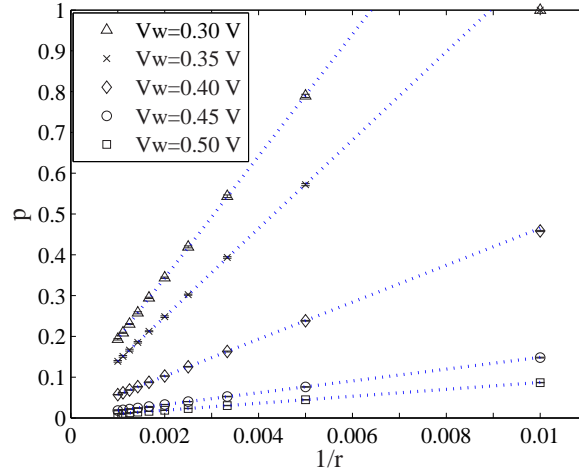


Figure 2.30: Mean probability as a function of the input spike rate for $V_w = 0.30, 0.35, 0.40, 0.45, 0.50$ V. The dotted lines show the least mean square fit for input rates from 200 Hz to 1000 Hz.

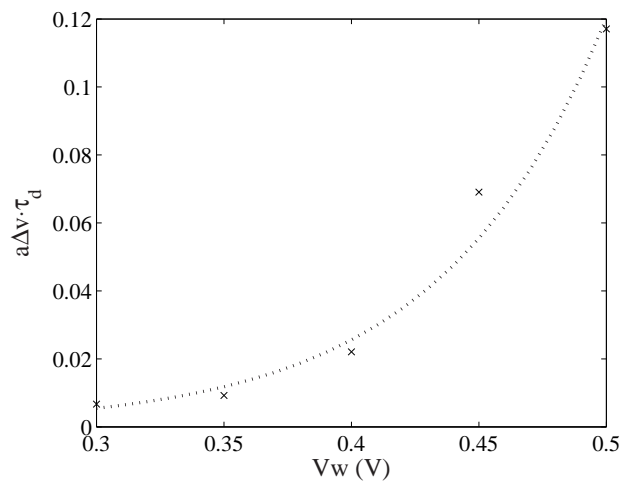


Figure 2.31: $a\Delta v\tau_d$ as a function of V_w . The dotted line shows the least mean square fit, $f(x) = e^{(15.47x-9.854)}$.

Chapter 3

Stochastic Circuit Modeling and Simulation

3.1 Introduction

Noise plays a significant role in circuit performance and is an important factor to consider for circuit design. Commonly noise effects are analyzed in frequency domain in terms of power spectral density (PSD). For a linear system, output noise PSD is related to input noise PSD by its transfer function, i.e., total output noise PSD is the summation of all noise sources scaled by the power transfer functions of all intervening subsystems. Input referred noise is computed by dividing the total output noise by the system power transfer function for direct comparison with input signals, i.e. to compute SNR [42, 43]. These capabilities are provided by popular circuit simulators. Such frequency domain analysis, however, is not suitable for many circuits, particularly where large signal behavior or nonlinearity is an important characteristic, or where performance depends on transient sample paths and ensemble statistics as for the random number generator (RNG) circuit [97].

We propose to study the large signal circuit stochastic behavior caused by noise using stochastic differential equations (SDEs). Time domain transient analysis using the numerical solution of the SDEs shows how the signal evolves in the presence of noise fluctuations, and how those fluctuations influence the statistics of the signal samples at specific times. Analytical equations of the probability evolution can

also be written and the solution for steady state could be obtained in some cases. The method can also be extended to small signal models where the circuit stochastic behavior is determined at a small signal scale. Small signal stochastic models directly relates the device parameters and operation biases to the stochastic dynamics of the circuits, which will help during the circuit design for desired behavior. We use the method to investigate the stochastic behavior of the stochastic synapse or RNG circuit (since the circuits can be used as either the stochastic synapse or RNG, we refer the circuits with either name in this chapter) [103].

With the continuing semiconductor device scaling, power supply scaling will reach the stage where the noise becomes significant compared with signals in circuits. Even digital circuits would exhibit stochastic behavior. We apply the method to an CMOS inverter, a basic unit in digital circuits, to understand the stochastic behavior of digital circuits operating at very low supply voltages [104]. We derive the analytical result of the steady state from first principle, which matches the simulation results very well.

We will provide a quick review of the circuit noise and introduction of the stochastic differential equation before we introduce our stochastic circuit modeling. The detailed treatment of circuit noise can be found in [42, 43].

3.2 Circuit Noise

In circuit, noise is represented by unwanted current or voltage signals. It is characterized by either the voltage power spectral density (PSD) $v_n^2(f)$ in the unit

of V^2/Hz or the current PSD $i_n^2(f)$ in the unit of A^2/Hz . The root mean square (RMS) of the noise is given by the noise PSD and bandwidth [43],

$$v_{RMS} = \sqrt{\int_{f_L}^{f_H} v_n^2(f) df} \quad (3.1)$$

$$i_{RMS} = \sqrt{\int_{f_L}^{f_H} i_n^2(f) df} \quad (3.2)$$

In the following sections, PSD of single sided frequencies is used. The total noise PSD is the summation of the individual noise PSD if those noise are independent.

3.2.1 Thermal Noise

Thermal noise is caused by the random thermal motion of electrons, therefore it is directly proportional to absolute temperature T . It is not affected by the actual current flowing through the component. It is present in any linear passive resistors. In a resistor R , thermal noise can be represented by a series voltage generator with PSD [42, 43]

$$v_n^2(f) = 4kTR \quad (3.3)$$

or a shunt current generator with PSD [42, 43]

$$i_n^2(f) = \frac{4kT}{R} \quad (3.4)$$

The noise PSD is flat across all frequencies in the bandwidth and its amplitude distribution is Gaussian, therefore it is also called white noise. For MOSFET, the thermal noise is commonly modeled by an equivalent resistor $R = \frac{3}{2} \frac{1}{g_m}$ in saturation above threshold, and $R = \frac{1}{\mu C_{ox} \frac{W}{L} (V_{gs} - V_{th} - V_{ds})}$ in triode region above threshold [43].

3.2.2 Shot Noise

Shot noise is caused by the discrete movement of charges across a potential barrier in the devices such as diodes, MOS transistor, and bipolar transistors. The apparent current I is composed by a large number of random independent current pulses, fluctuating around its mean value. The current noise PSD is [42]

$$i_n^2(f) = 2qI \quad (3.5)$$

It is a Gaussian white noise as well.

3.2.3 Flicker Noise

Flicker noise is also called 1/f noise because its PSD has a 1/f dependence [42],

$$i_n^2(f) = K \frac{I^a}{f^a} \quad (3.6)$$

where I is the direct current in the device, K , a , b are device dependent constants. It is caused by the electrons captured and released from the traps associated with contamination and crystal defect. The amplitude distribution is non-Gaussian. For MOSFET, flicker noise results from the trap at the Si-SiO₂ interface. Its PSD is given by [43]

$$i_n^2(f) = \frac{K_F I_D^{A_F}}{f(C_{ox})^2 W L} \quad (3.7)$$

where K_F is the flicker noise coefficient, I_D is the channel current, A_F is the flicker noise exponent, C_{ox} is the silicon-dioxide capacitance per unit area, and W , L are the gate width and length.

The three types of noise just introduced contribute the most noise in regular circuits. For some special circuits, other noise could be dominant. For completeness, they are listed here as well.

3.2.4 Burst Noise

Burst noise is related to the presence of heavy-metal ion contamination. Most of its power is also in low-frequency, similar to the $1/f$ noise. The PSD is [42]

$$i_n^2(f) = K \frac{I^c}{1 + \left(\frac{f}{f_c}\right)^2} \quad (3.8)$$

where I is the direct current, K and c are device dependent constants. f_c determines the bandwidth of the noise. Burst noise is a non-Gaussian noise.

3.2.5 Avalanche Noise

Avalanche noise is produced by avalanche breakdown in pn junction. Holes and electrons with enough energy in the depletion region of the reverse-biased pn junction create new hole-electron pairs, and these hole-electron pairs create even more pairs in a cumulative process. The noise produces large current fluctuations.

3.3 Stochastic Differential Equations

In this section, I will introduce the basic mathematical concepts and tools for stochastic modeling and simulation. Detailed treatment can be found in [105–107].

3.3.1 Random Variable and Stochastic Process

An experiment that generates random outcomes can be described by a probability space, the triplet (Ω, \mathcal{A}, P) . The sample space Ω is a set consisting of all possible outcomes of the experiment. The \mathcal{A} is a σ -algebra that consists of events, defined as subsets of Ω . The probability measure $P : \mathcal{A} \rightarrow [0, 1]$ gives the probability of the events in \mathcal{A} [105]. A random variable $X : \Omega \rightarrow \mathbb{R}$ maps each outcome $\omega \in \Omega$ to a real value $X(\omega)$, and $X^{-1}(B) \in \mathcal{A}$ for all $B \in \mathcal{B}$ where \mathcal{B} is the Borel σ -algebra of \mathbb{R} . The function $F_X(x) = P(\omega \in \Omega : X(\omega) \leq x)$ is called the probability distribution of the random variable X . For a continuous random variable, a probability density function $p(x)$ is defined such that $F_X(x) = \int_{-\infty}^x p(s)ds$ [105]. A Hilbert space of random variables H_{RV} is a complete inner product space of the random variables defined on a probability space. The inner product of two random variable X and Y in the space is defined as $\langle X, Y \rangle = E(XY)$, where $E(\cdot)$ is the expectation. And the norm of the random variable X is $\|X\| = \langle X, X \rangle^{\frac{1}{2}}$ [105].

A stochastic process is a family of random variables $X(t)$ defined on a probability space (Ω, \mathcal{A}, P) . It is a function of two variables t and ω : $\tau \times \Omega \rightarrow \mathbb{R}$. For each $t \in \tau$, $X(t) = X(t, \cdot)$ is a random variable, while for each $\omega \in \Omega$, $X(\cdot, \omega)$ gives a sample path or a trajectory of the stochastic process [105]. Index t can be discrete or continuous. ω is generally suppressed in the expression of a random process. If the future state of the process is only determined by current state other than the past state, i.e., $P(X(t_{n+1})|X(t_n) = x_n) = P(X(t_{n+1})|X(t_1) = x_1, \dots, X(t_n) = x_n)$, the process is called a Markov process. The Hilbert space of stochastic processes

H_{SP} on the interval $[0, T]$ and (Ω, \mathcal{A}, P) is a complete inner product space with $\langle X, Y \rangle = \int_0^T E(X(t)Y(t))dt$ and norm $\|X\| = (\int_0^T E|X(t)|^2 dt)^{1/2}$ [105].

Let $X_1(t), \dots, X_n(t)$ be n i.i.d. Poisson processes with rate λ . By Central Limit Theorem, $Y_n(t) = \sum_{i=1}^n \frac{X_i(t) - \lambda t}{\sqrt{\lambda n}}$ approaches a normally distributed random variable $N(0, t)$ as $n \rightarrow \infty$. The process $Y_n(t)$ actually approaches a Wiener process or Brownian motion $W(t)$ [105]. A standard Wiener process $W(t)$ has the properties [105–107]:

- It has independent, stationary, and Gaussian increment;
- $E(W(t)) = 0$
- $Var(W(t) - W(s)) = E[(W(t) - W(s))^2] = t - s$
- $W(0) = 0$
- $W(t)$ is continuous in the mean square sense
- $W(t)$ is nowhere differentiable

Wiener process is a continuous homogeneous Markov process. Its transition probability density $p(y, t, x, s) = p(W(t) = y | W(s) = x), s < t$ is given by [105]

$$p(y, t, x, s) = \frac{1}{\sqrt{2\pi(t-s)}} \exp\left(\frac{-(x-y)^2}{2(t-s)}\right) \quad (3.9)$$

A Wiener process sample path can be generated at a finite number of points $0 = t_0 < t_1 < \dots < t_N = T$ on the interval $[0, T]$ by [105, 106]

$$W(t_i) = W(t_{i-1}) + \sqrt{t_i - t_{i-1}} X_{i-1} \quad (3.10)$$

where $X_{i-1} \sim N(0, 1)$ are independent normally distributed numbers. A continuous approximation to the Wiener process can then be obtained by a piecewise linear stochastic process [105]

$$Y_n(t) = W(t_i) \frac{t_{i+1} - t}{h} + W(t_{i+1}) \frac{t - t_i}{h} \quad (3.11)$$

for $t_i \leq t \leq t_{i+1}$ and $i = 0, 1, \dots, N - 1$. It can be shown that $\lim_{n \rightarrow \infty} \|Y_n - W\| = 0$ [105].

3.3.2 Stochastic Integrals

Integrals of the form $\int_a^b f(s, \omega) ds$ and $\int_a^b f(s, \omega) dW(s, \omega)$ are introduced here and will be used in the following section of stochastic differential equations. $f(s, \omega)$ is a stochastic process on $[a, b] \times (\Omega, \mathcal{A}, P)$, satisfying the following conditions [105]:

C1: $\|f(a)\|^2 \leq k_1$, k_1 is a positive constant

C2: $\|f(t_2) - f(t_1)\|^2 \leq k_2 |t_2 - t_1|$ for $\forall t_1, t_2 \in [a, b]$, k_2 is a positive constant

C3: f is nonanticipating on $[a, b]$

For clarity, ω is kept in the expression in this section.

3.3.2.1 Integral $J(f)(\omega) = \int_a^b f(s, \omega) ds$

$\int_a^b f(s, \omega) ds$ is an integration of a stochastic process along time. Its definition is similar to the Reimann integral for deterministic functions. Let $f^{(n)}(t, \omega) = \sum_{i=0}^{n-1} f(t_i, \omega) I_i^{(n)}(t)$ be a sequence of step functions, where $a = t_0^{(n)} < \dots < t_n^{(n)} = b$

is a family of partitions of $[a, b]$, $\max_{0 \leq i \leq n-1} t_{i+1}^{(n)} - t_i^{(n)} \rightarrow 0$ as $n \rightarrow \infty$. $I_i^{(n)}(t)$ is an indicator function

$$I_i^{(n)}(t) = \begin{cases} 1 & \text{for } t_i^{(n)} \leq t < t_{i+1}^{(n)} \\ 0 & \text{otherwise} \end{cases} \quad (3.12)$$

Since $f^{(n)}$ converges to f in the mean square sense, the integral $\int_a^b f(s, \omega) ds$ is defined as the mean square limit of [105]

$$J(f^{(n)})(\omega) = \int_a^b f^{(n)}(s, \omega) ds = \sum_{i=0}^{n-1} f(t_i^{(n)}, \omega)(t_{i+1}^{(n)} - t_i^{(n)}) \quad (3.13)$$

as $n \rightarrow \infty$. $J(f)(\omega)$ is a random variable. $J(f)(t, \omega) = \int_a^t f(s, \omega) ds$ is defined in the similar way, which is a random process. The integration can be approximated by $J_n(f) = \sum_{i=0}^{n-1} f(t_i) \Delta$, where $t_i = a + i\Delta$, $\Delta = (b-a)/n$. $\|J(f) - J_n(f)\| = O(1/\sqrt{n})$ [105].

3.3.2.2 Ito Integral $I(f)(\omega) = \int_a^b f(s, \omega) dW(s, \omega)$

$\int_a^b f(s, \omega) dW(s, \omega)$ is an integration of a stochastic process relative to a Wiener process. Let $a = t_0^{(n)} < \dots < t_n^{(n)} = b$ be a family of partitions of $[a, b]$, where $\max_{0 \leq i \leq n-1} t_{i+1}^{(n)} - t_i^{(n)} \rightarrow 0$ as $n \rightarrow \infty$. $f^{(n)}$ is a step function defined as $f^{(n)}(t, \omega) = f(t_j^{(n)}, \omega)$ on $t \in [t_j^{(n)}, t_{j+1}^{(n)})$ where $j = 0, 1, \dots, n-1$. For each sample path, the Ito definition of integral of the form $\int_a^b f(s, \omega) dW(s, \omega)$ is the mean square limit of [105]

$$I(f^{(n)})(\omega) = \int_a^b f^{(n)}(s, \omega) dW(s, \omega) = \sum_{j=0}^{n-1} f^{(n)}(t_j^{(n)}, \omega)(W(t_{j+1}^{(n)}, \omega) - W(t_j^{(n)}, \omega)) \quad (3.14)$$

as $n \rightarrow \infty$. Similarly, $I(f)(t, \omega) = \int_a^t f(s, \omega) dW(s, \omega)$ can be defined. Ito integral can be approximated by $I_n(f) = \sum_{i=0}^{n-1} f(t_i)(W(t_{i+1}) - W(t_i))$, where $t_i = a + i\Delta$

and $\Delta = (b - a)/n$. $\|I(f) - I_n(f)\| = O(1/\sqrt{n})$ [105].

3.3.3 Stochastic Differential Equation

Consider a stochastic process satisfying the stochastic differential equation:

$$dX(t, \omega) = f(t, X(t, \omega))dt + g(t, X(t, \omega))dW(t, \omega) \quad (3.15)$$

f is called the drift coefficient and g is called the diffusion coefficient. Both f and g are nonanticipating functions. The solution exists for (3.15) if with probability one, a solution exists for any choice of sample functions of the Wiener process $W(t)$. The solution is unique if for a given sample function of $W(t)$, the particular solution of the equation which arises is unique. The sufficient conditions for existence and uniqueness of solution $X \in H_{SP}$ in a time interval $[a, b]$ are that both f and g satisfy [107]:

C4 (Lipschitz condition): $\exists K, |f(t, x) - f(t, y)| + |g(t, x) - g(t, y)| \leq K|x - y|$, for all x, y , and $a \leq t \leq b$.

C5 (growth condition): $\exists K, |f(t, x)|^2 + |g(t, x)|^2 \leq K^2(1 + |x|^2)$ for all x and $a \leq t \leq b$.

The solution can be written in the integral formula as

$$X(t, \omega) = X(a, \omega) + \int_a^t f(s, X(s, \omega))ds + \int_a^t g(s, X(s, \omega))dW(s, \omega) \quad (3.16)$$

Let $Y : [0, T] \times \mathbb{R} \rightarrow \mathbb{R}$ have continuous first and second order derivatives, the Ito integral of $Y(t, x(t))$ is given by Ito formula [106, 107]:

$$Y(t, x_t) = Y(t_0, x_{t_0}) + \int_{t_0}^t \left(\frac{\partial Y}{\partial t} + f(s, x_s) \frac{\partial Y}{\partial x} + \frac{1}{2} g(s, x_s)^2 \frac{\partial^2 Y}{\partial x^2} \right) ds + \int_{t_0}^t g(s, x_s) \frac{\partial Y}{\partial x} dW_s \quad (3.17)$$

3.3.4 Numerical Solution of Stochastic Differential Equation

Numerical solution of the SDE can be obtained by various discrete time approximations to the real solution of the SDE [106]. Several methods are based on discrete time approximations with truncated Ito-Taylor expansions. Applying Ito formula to $f(t, x)$ and $g(t, x)$ in (3.16), we obtain Ito-Taylor expansion of x_t . It can have arbitrarily many expansion terms depending on how many times Ito formula is applied. Limiting ourselves to the case where f and g are not explicit functions of t , we obtain two simple and commonly used expansions [106]:

$$x_t = x_{t_0} + f(x_{t_0}) \int_{t_0}^t ds + g(x_{t_0}) \int_{t_0}^t dW_s + R1 \quad (3.18)$$

$$x_t = x_{t_0} + f(x_{t_0}) \int_{t_0}^t ds + g(x_{t_0}) \int_{t_0}^t dW_s + b(x_{t_0})b'(x_{t_0}) \int_{t_0}^t \int_{t_0}^s dW_z dW_s + R2 \quad (3.19)$$

where $R1$ and $R2$ are the remainders. Ignoring the remainder and using the expansion as the discrete time approximation we get numerical approximation of x_t . Let \hat{x}_n be the discrete time approximation of $x(t_n)$, where $t_n = t_0 + n \cdot \Delta$, Δ is the time step, the Euler method [106] gives

$$\hat{x}_{n+1} = \hat{x}_n + a(\hat{x}_n)\Delta + b(\hat{x}_n)\Delta W_n \quad (3.20)$$

and the Milstein method [106] gives

$$\hat{x}_{n+1} = \hat{x}_n + a(\hat{x}_n)\Delta + b(\hat{x}_n)\Delta W_n + \frac{1}{2}b(\hat{x}_n)b'(\hat{x}_n)((\Delta W_n)^2 - \Delta) \quad (3.21)$$

ΔW_n is a Gaussian random variable with distribution $N(0, \Delta)$. It can be generated by a software RNG for simulation.

The performance of the approximation is quantified by different criterion depending on the goal of the approximation. For a pathwise approximation, strong convergence is required, where $E(|x_t - \hat{x}_t|) \rightarrow 0$ as the step size goes to zero. A discrete time approximation converges strongly with order $\gamma > 0$ at time t if there exists a positive constant C , which does not depend on Δ , and a finite δ such that $E(|x_t - \hat{x}_t|) \leq C\Delta^\gamma$ for each $\Delta \in (0, \delta)$ [106]. The Euler method has a strong order of 0.5, while the Milstein method has a strong order of 1 [106]. For approximation of moments, probability, or other expectations of functionals of the stochastic process, the requirement for simulation is less demanding, and a weaker convergence criterion is used, where $|E(g(x_t)) - E(g(\hat{x}_t))| \rightarrow 0$ as $\Delta \rightarrow 0$ for all g in a class of functions. It has weak convergence order β if for each polynomial g there exists a positive constant C , which does not depend on Δ , and a finite δ such that $|E(g(x_t)) - E(g(\hat{x}_t))| \leq C\Delta^\beta$ for each $\Delta \in (0, \delta)$ [106]. So far, all the development is for single variant SDEs. Similar approach can be applied to vector SDEs.

3.3.5 Fokker-Planck Equation

For the stochastic process described by (3.15), the time evolution of the its probability density function $p(x, t)$ satisfies the Fokker-Planck equation:

$$\frac{\partial p(x, t)}{\partial t} = -\frac{\partial}{\partial x}[f(x, t)p(x, t)] + \frac{1}{2} \frac{\partial^2}{\partial x^2}[g^2(x, t)p(x, t)] \quad (3.22)$$

with the initial condition $p(x, t)|_{t=t_0} = p(x, t_0)$ [107].

SDE has been applied in many fields such as population biology, finance, chemical reactions, mechanic systems, to model the stochastic behavior caused by internal or external fluctuations [105]. Next, I will apply SDE to model the circuit stochastic behavior driven by noise.

3.4 Stochastic Circuit Model

I first introduce a stochastic model for a voltage node in the circuit to illustrate the idea of stochastic circuit model, then extend it to two circuits.

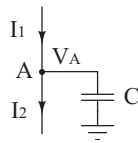


Figure 3.1: Schematic of a simple voltage node.

3.4.1 Voltage Node Stochastic Model

Fig. 3.1 shows the simplest case of a node A where there is an incoming current to and an outgoing current from the node, and a capacitor connected to

the node. The capacitor could be either an explicit capacitor or some parasitic capacitance associated with devices in the circuit. In our model, noise is considered in current mode, and only white noise is considered. Similar methods can extend such analysis to more complicated nodes. We write down the ordinary differential equation (ODE) of the voltage at node A:

$$C \frac{dV_A}{dt} = I_1 - I_2 + \xi \quad (3.23)$$

where ξ is a rapidly fluctuating random term (an implicit current not shown in Fig. 3.1), modeled as an idealized Gaussian white current noise: for any $t \neq s$, $\xi(t)$ and $\xi(s)$ are statistically independent, and the autocorrelation of $\xi(t)$ is $E(\xi(s)\xi(t)) = \delta(s - t)$. The mean of the noise $E(\xi(t)) = 0$ for any t . $\xi(t)$ can be viewed as the derivative of a Brownian motion or a Wiener process [106, 107]:

$$\xi(t) = \sqrt{R(t)} \frac{dW(t)}{dt} \quad (3.24)$$

where $W(t)$ is a standard Wiener process with the following properties: 1) independent, Gaussian, and stationary increments; 2) $E(W(t)) = 0$; 3) $W(0) = 0$; 4) $E[(W(t) - W(s))^2] = t - s$. $R(t)$ is the power spectral density (PSD) of the Gaussian white current noise: it is flat across all frequencies and its amplitude is determined by the circuit elements connected to node A. $R(t)$ is constant for stationary white noise. A non-stationary white noise can be viewed as a stationary white noise modulated by a time-varying function $R(t)$. Now we can write the SDE for V_A as:

$$dV_A(t) = \frac{I_1(t) - I_2(t)}{C} dt + \frac{\sqrt{R(t)}}{C} dW(t) \quad (3.25)$$

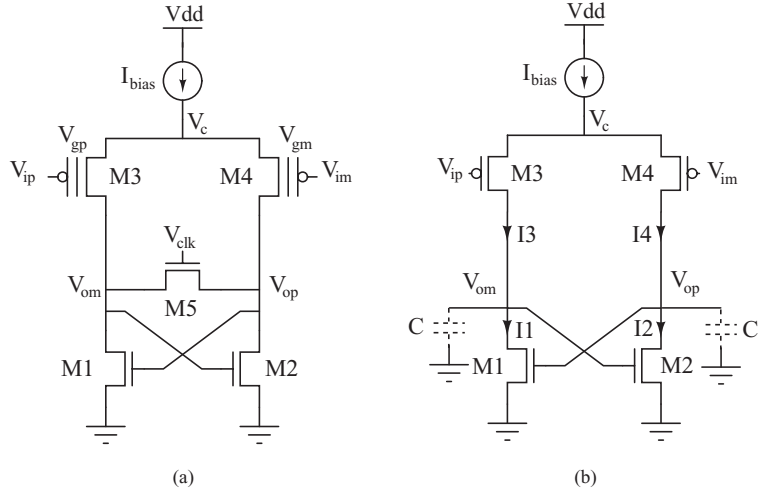


Figure 3.2: Stochastic synapse circuits: a) the clocked cross-coupled differential pair comparator in the stochastic synapse circuit, b) the simplified circuit for stochastic modeling, parasitic capacitances are explicitly added.

3.4.2 Stochastic Synapse Circuit

Although the previous chapter presented an intuitive explanation of how the stochastic synapse circuit in Fig. 3.2(a) works, a quantitative model is necessary to fully understand how noise interacts with the circuit dynamics. Here we develop the stochastic model for the stochastic synapse circuit using SDEs.

3.4.2.1 Stochastic Model of the Stochastic Synapse Circuit

Fig. 3.2(b) is the simplified circuit for the stochastic model, where floating-gate pFETs are replaced with regular pFETs assuming that they are matched, and the switch transistor M5 is removed because we are considering the stochastic behavior once the switch is open. Parasitic capacitors C are added on each side. We operate the circuit in subthreshold or weak inversion region. Previous work [108] have

shown that the major drain current noise of MOSFET in subthreshold is white and Gaussian. The noise could be viewed as either shot noise or thermal noise [108–110]. The expression for shot noise and thermal noise are unified in subthreshold region. Wyatt and Coram examined nonlinear device noise models and found that only nonlinear shot noise models predicts thermodynamically acceptable behavior [111]. Therefore we use the shot noise expression qI_d as the PSD of the noise current in subthreshold MOSFET, where I_d is the drain current. The SDEs of V_{om} and V_{op} are two SDEs coupled together:

$$dV_{om} = \frac{1}{C}(I_3(V_{om}) - I_1(V_{om}, V_{op}))dt + \frac{1}{C}\sqrt{q(I_3(V_{om}) + I_1(V_{om}, V_{op}))}dW_1 \quad (3.26)$$

$$dV_{op} = \frac{1}{C}(I_4(V_{op}) - I_2(V_{om}, V_{op}))dt + \frac{1}{C}\sqrt{q(I_4(V_{op}) + I_2(V_{om}, V_{op}))}dW_2 \quad (3.27)$$

We use EKV model for the drain current I_1 to I_4 because it provides a smooth transition from subthreshold (exponential function) to above threshold (quadratic function), thus a better modeling of weak inversion [112]. The I-V relation in EKV model for nFET and pFET are

$$I_n = \frac{W}{L}2V_T^2\frac{\mu_n C_{ox}}{\kappa}(\ln^2(1 + e^{\frac{\kappa(V_{gb}-V_{thn})-V_{sb}}{2V_T}}) - \ln^2(1 + e^{\frac{\kappa(V_{gb}-V_{thn})-V_{db}}{2V_T}})) \quad (3.28)$$

$$I_p = \frac{W}{L}2V_T^2\frac{\mu_p C_{ox}}{\kappa}(\ln^2(1 + e^{\frac{\kappa(V_{bg}-|V_{thp}|-V_{bs}}{2V_T}}) - \ln^2(1 + e^{\frac{\kappa(V_{bg}-|V_{thp}|-V_{bd}}{2V_T}})) \quad (3.29)$$

We simulate the circuit starting from the initial metastable state right after the switch is open. We use the parameters that closely match our fabricated chip: either from the layout or the test data of the run provided by Mosis. We use the Euler method because it has both strong and weak convergence, and it is computationally cheap. Fig. 3.3 shows one simulation trajectory at bias current 1 μ A, V_{om} and

V_{op} diverges as we expected. Fig. 3.4 shows the zoom-in picture at the beginning, where the two voltages fluctuate and intersect each other for a couple of times before diverging eventually. Fig. 3.5 shows ten runs where sometimes V_{om} goes high and sometimes V_{op} goes high. We then sweep the input offset $V_{ip} - V_{im}$ from -8mV to 8mV. For each input offset we simulate the circuit 10^4 times and calculate the probability $p(V_{op} > V_{om})$. We plot the probability as a function of input offset in Fig. 3.6. Similar to the experimental result, the function is closely fitted by an error function $f(v) = 0.5 \left(1 + \text{erf} \left(\frac{v-\mu}{\sqrt{2}\delta} \right) \right)$, where $\mu = 0.0157$ and $\delta = 1.2659$. We perform the same simulation for the bias current from $100 \mu\text{A}$ to 10 pA . Fig. 3.7 shows that for $I_{bias} \geq 1 \mu\text{A}$, increasing I_{bias} widens the probability tuning, while for $I_{bias} \leq 100 \text{ nA}$, I_{bias} has little influence on the probability tuning.

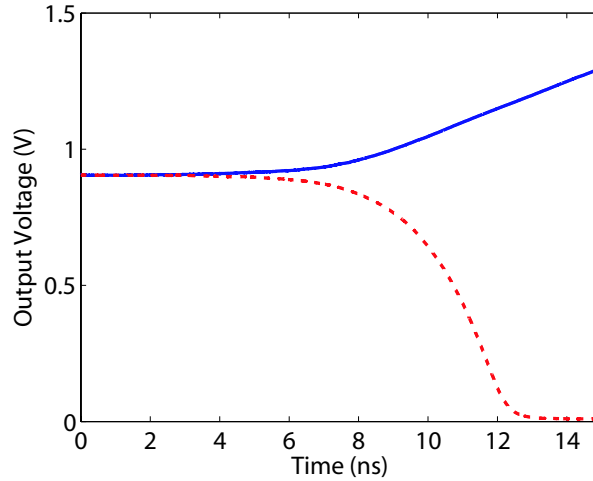


Figure 3.3: One simulated sample path of the stochastic synapse, dashed line: V_{op} , solid line: V_{om} .

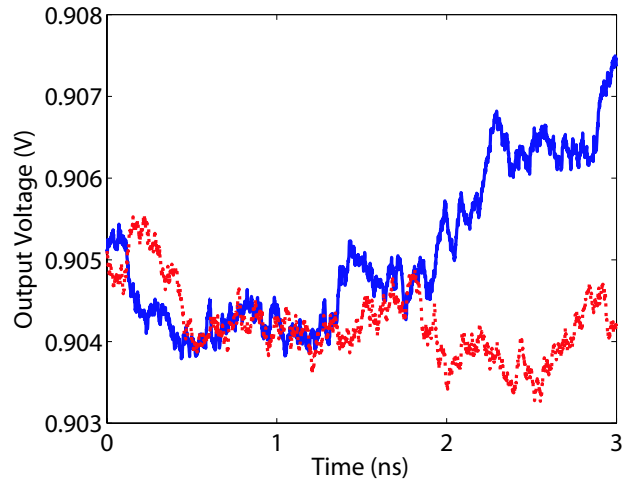


Figure 3.4: Close-up view of the initial period of Fig. 3.3, dashed line: V_{op} , solid line: V_{om} .

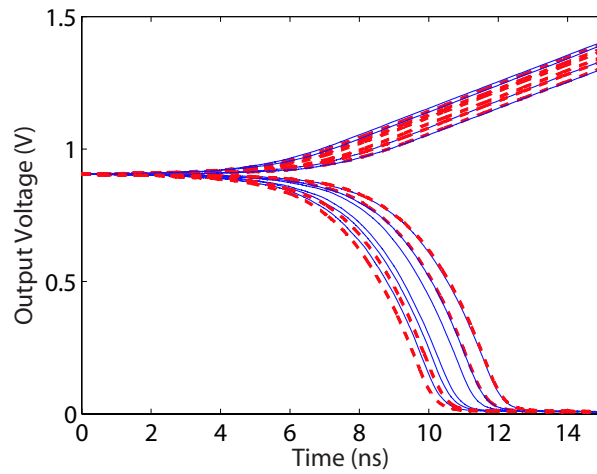


Figure 3.5: Multiple sample paths from 10 transient simulations, dashed line: V_{op} , solid line: V_{om} .

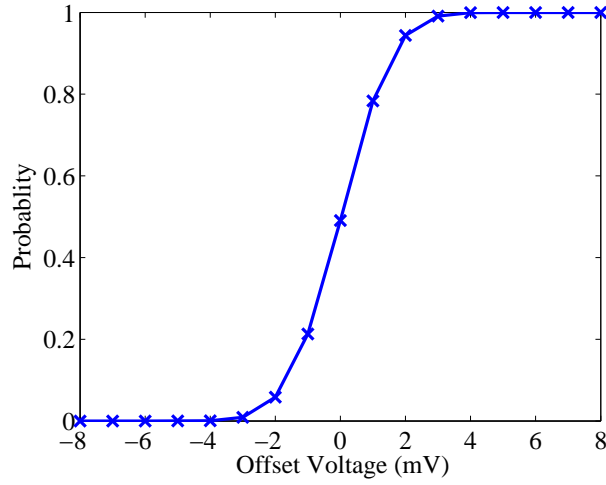


Figure 3.6: The probability $p(V_{op} > V_{om})$ as a function of input offset $v = V_{ip} - V_{im}$. The solid line shows the fitting of an error function $f(v) = 0.5 \left(1 + \operatorname{erf} \left(\frac{v-\mu}{\sqrt{2}\delta} \right) \right)$, where $\mu = 0.0157$ and $\delta = 1.2659$.

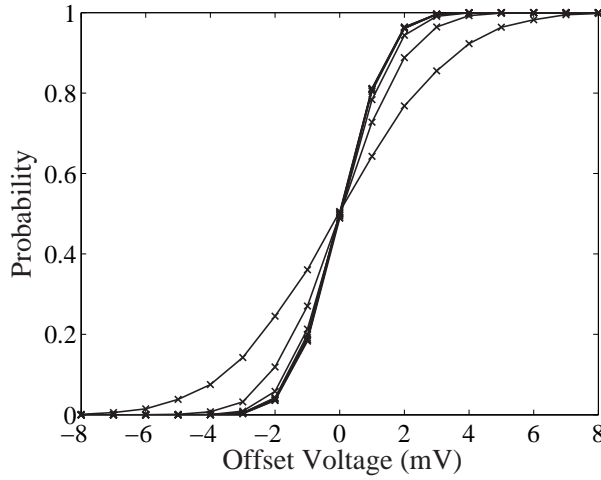


Figure 3.7: The probability $p(V_{op} > V_{om})$ as a function of input offset $V_{ip} - V_{im}$ for $I_{bias} = 100 \mu\text{A}, 10 \mu\text{A}, 1\mu\text{A}, 100 \text{ nA}, 10 \text{ nA}, 1 \text{ nA}, 100 \text{ pA},$ and 10 pA . $\delta=2.8039, 1.6547, 1.2659, 1.1666, 1.1361, 1.1402, 1.1216,$ and 1.1246 respectively, for the fitting of the error function.

3.4.2.2 Small Signal Stochastic Model

To further investigate how the device parameters and bias condition affect the performance of the stochastic synapse, we write down the small signal model of the circuits right after the switch is open, as shown in Fig. 3.8. Since there is a

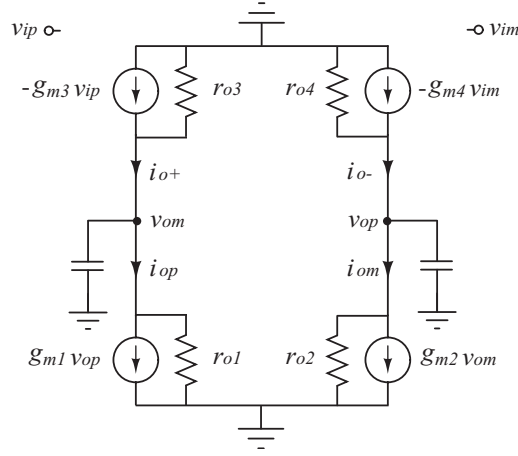


Figure 3.8: Small signal stochastic model of the stochastic synapse circuit.

regenerative positive feedback in the circuit because of the cross coupling, normal small signal analysis would not work. Therefore we separate the circuit into two parts, the differential pair and the cross coupling; write down their small signal models respectively; then combine them together to obtain the small signal model for the entire circuit. The small signal model for the differential pair is

$$i_{o+} = g_{m3}(-v_{ip}) + \frac{1}{r_{o3}}(-v_{om}) \quad (3.30)$$

$$i_{o-} = g_{m4}(-v_{im}) + \frac{1}{r_{o4}}(-v_{op}) \quad (3.31)$$

And the small signal model for the cross coupling is

$$i_{op} = g_{m1}v_{op} + \frac{1}{r_{o1}}v_{om} \quad (3.32)$$

$$i_{om} = g_{m2}v_{om} + \frac{1}{r_{o2}}v_{op} \quad (3.33)$$

If we bias the input voltage V_{ip} and V_{im} close enough, $g_{m3} \approx g_{m4}$, $g_{m1} \approx g_{m2}$, $r_{o1} \approx r_{o2}$, $r_{o3} \approx r_{o4}$. Since $g_{m3} \gg \frac{1}{r_{o3}}$ and $g_{m1} \gg \frac{1}{r_{o1}}$, we have

$$i_{o+} - i_{o-} \approx -g_{m3}(v_{ip} - v_{im}) \quad (3.34)$$

$$i_{op} - i_{om} \approx g_{m1}(v_{op} - v_{om}) \quad (3.35)$$

Substitute them into the differential equation of v_{om} and v_{op}

$$Cdv_{om} = (i_{o+} - i_{op})dt \quad (3.36)$$

$$Cdv_{op} = (i_{o-} - i_{om})dt \quad (3.37)$$

We obtain

$$Cdv = g_{m3}V_{os}dt + g_{m1}vdt \quad (3.38)$$

where v is the differential output voltage $v_{op} - v_{om}$, and $V_{os} = v_{ip} - v_{im} = V_{ip} - V_{im}$ is the input offset voltage. The initial condition of (3.38) is $v(0) = 0$. Adding noise component into the equation, we obtain the small signal SDE:

$$Cdv = (g_{m3}V_{os} + g_{m1}v)dt + \sqrt{R}dW \quad (3.39)$$

where we use white Gaussian current noise, $R = 2qI_{bias}$.

Compared with the previous SDE model, this model is much simpler, highlighting the direct relation between the device parameters and the circuit dynamics.

A close inspection of the SDE shows that $g_{m3}V_{os}$ is the initial offset that drives the SDE, and g_{m1} is the positive feedback gain. Without the noise, the direction of the divergence of v is deterministic, only determined by V_{os} . With the presence of noise, the noise term $\sqrt{R}dW$ can compete with the influence of the initial offset from the beginning until v diverges from 0 large enough for positive feedback to dominate, thus generates the stochastic divergence. And V_{os} determines the probability distribution of the outcome. Equation (3.38) can be further written as

$$Cdv = \left(\frac{\kappa I_{bias}}{2V_T} V_{os} + \frac{\kappa I_{bias}}{2V_T} v \right) dt + \sqrt{2qI_{bias}} dW \quad (3.40)$$

in subthreshold and

$$Cdv = \left(\sqrt{\beta_3 I_{bias}} V_{os} + \sqrt{\beta_1 I_{bias}} v \right) dt + \sqrt{2qI_{bias}} dW \quad (3.41)$$

approximately in above threshold, where $\beta_3 = \mu_p C_{ox} \frac{W_3}{L_3}$ and $\beta_1 = \mu_n C_{ox} \frac{W_1}{L_1}$. We simulated the SDE numerically using the same parameters as before in different regions and obtained similar stochastic behavior. The probability of v diverging to a positive value matches well with the probability $p(V_{op} > V_{om})$ obtained from the large signal model simulation, demonstrating the similar influence of bias current where the probability tuning in subthreshold does not change with bias, whereas in above threshold operation, tuning is wider with higher bias current, as shown in both Fig. 3.7 and Fig. 3.9. This model also shows that in the subthreshold operation, the size of the transistors does not affect the probability tuning, whereas in above threshold operation, decreasing $W_{3,4}/L_{3,4}$ widens the probability tuning. This insight is confirmed by simulation. With $W_{1,2}/L_{1,2} = 10/10$ in above threshold, when $W_{3,4}/L_{3,4} = 10/10$, $\delta = 2.049$, while when $W_{3,4}/L_{3,4} = 2/10$, $\delta = 4.609$.

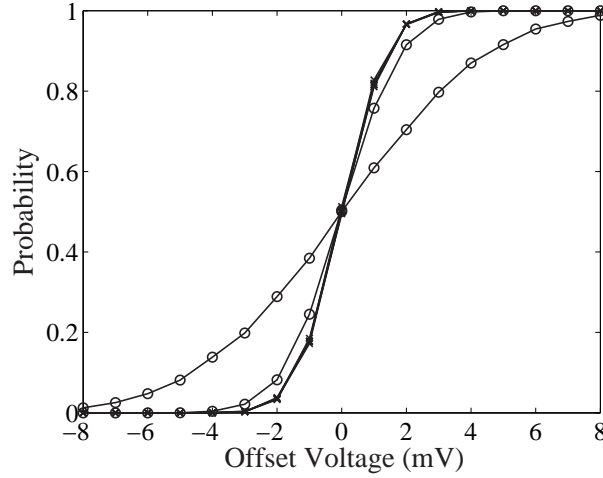


Figure 3.9: The probability $p(V_{op} > V_{om})$ as a function of input offset $V_{ip} - V_{im}$, \circ : from the above threshold small signal model (3.41) for $I_{bias} = 100 \mu\text{A}$, $10 \mu\text{A}$, \times : from the subthreshold small signal model (3.40) for $I_{bias} = 10 \text{ nA}$, 1 nA , 100 pA , 10 pA , and 1 pA . $\delta = 3.5990, 1.4475, 1.0776, 1.0980, 1.1098, 1.0972$, and 1.0903 for the fitting of the error function.

We measured the probability tuning curve of the circuits with two different geometrical sizes under various bias currents. As shown in Table 3.1, when I_{bias} is small so that the circuits operate in subthreshold, there is little difference of the probability tuning between two layouts; when I_{bias} increases so that the circuits operate above threshold, the higher the I_{bias} , the wider the probability tuning, and the transistor sizes of $M3$ and $M4$ affect the probability tuning, the smaller the $W_{3,4}/L_{3,4}$ is, the wider the probability tuning. These results match the prediction from the small signal stochastic model.

From the SDE, we can write down the evolution of the conditional probability

Table 3.1: Comparison of δ in probability tuning

Layout	I_{bias}				
	10 μ A	1 μ A	100 nA	10 nA	1 nA
$\frac{W_{3,4}}{L_{3,4}} = \frac{4}{12}, \frac{W_{1,2}}{L_{1,2}} = \frac{10}{10}$	2.2711	0.8436	0.3898	0.3054	0.2914
$\frac{W_{3,4}}{L_{3,4}} = \frac{14}{2}, \frac{W_{1,2}}{L_{1,2}} = \frac{10}{10}$	0.4344	0.3284	0.2628	0.2817	0.3205

$p(v, t|v_0, t_0)$ using Fokker-Planck equation

$$\frac{\partial P(v, t)}{\partial t} = -\frac{\partial((g_{m3}V_{os} + g_{m1}v)P(v, t))}{\partial v} + qI_{bias}\frac{\partial^2 P(v, t)}{\partial v^2} \quad (3.42)$$

where $P(v, t) = p(v, t|v_0, t_0)$. The initial condition of the equation is $P(v, 0) = \delta(0)$. Substitute (3.40) and (3.41) in (3.42) we obtain the Fokker-Planck equation for the subthreshold operation:

$$\frac{\partial P(v, t)}{I_{bias}\partial t} = -\frac{\partial(\frac{\kappa}{2V_T}(V_{os} + v)P(v, t))}{\partial v} + q\frac{\partial^2 P(v, t)}{\partial v^2} \quad (3.43)$$

and the above threshold operation:

$$\frac{\partial P(v, t)}{\sqrt{I_{bias}}\partial t} = -\frac{\partial((\sqrt{\beta_3}V_{os} + \sqrt{\beta_2}v)P(v, t))}{\partial v} + q\sqrt{I_{bias}}\frac{\partial^2 P(v, t)}{\partial v^2} \quad (3.44)$$

In (3.43), I_{bias} determines the time constant of the differential equation and would not affect the steady state distribution. This agrees with the simulation and experimental result that I_{bias} has little effect of the probability tuning in the subthreshold. In 3.44, I_{bias} has more complicated involvement in the dynamics of $P(v, t)$ for the above threshold operation.

$P(v, t)$ starts as an impulse at 0, then spreads and forms two peaks that move away from 0 toward both positive and negative values on v axis.

3.4.3 CMOS Inverter

Progress in semiconductor integrated circuits technology is driven largely by technical challenges associated with the continued scaling down of device sizes. In order to keep the internal electrical field from becoming unreasonably high at reduced device dimensions, power supply voltages have to be scaled accordingly. Low power supply voltages also introduce significant design constraints in low power applications.

The power supply range defines the dynamic range of the signals that can be represented and processed by a circuit. Lower power supply provides smaller signal range. When the signal range is so small that it is close to the magnitude of noise, signals are degraded significantly by noise. This signal corruption affects not only analog circuits but digital circuits as well. Logic high and logic low are normally represented by voltage signals higher or lower than a threshold voltage. When the signal voltage range is small, the signal amplitude can fluctuate back and forth across the threshold, resulting in an ambiguous logic value. This poses serious challenges to the conventional paradigm of deterministic computation based on digital logic circuits. New computational paradigms that account for this randomness and take advantage of it may achieve robust performance despite the noise [3–8]. These are different efforts and approaches to the circuit stochastic behavior from the biologically inspired stochastic synapse presented in the previous chapter.

The CMOS inverter is a fundamental circuit building block for digital circuits. Understanding the stochastic behavior of the CMOS inverter is essential to under-

standing the impact of noise on logic circuits at low power supply voltages in general. The inverter has been used as a model component to investigate minimum switching energy [113], bit-energy and information capacity [114], and energy-probability relation [115]. Using the method we proposed for stochastic circuit modeling and transient analysis, We develop a stochastic model of the inverter operating at low power supply, simulate the time domain transient response of the inverter to input logic switching, and obtain steady state statistics from multiple simulations. We determine the probability evolution equation, the Fokker-Planck equation, for the output voltage, and solve it analytically. The result from the solution matches closely with our simulation result. Extending the investigation of the CMOS inverter started by Abshire [114], this study provides a unique characterization of the stochastic behavior of circuit elements operating at low power, therefore might help us to find solutions for the usage of the stochastic properties of conventional digital circuits.

3.4.3.1 Dynamics of an Ideal CMOS Inverter

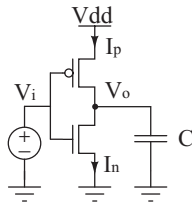


Figure 3.10: A CMOS inverter with a capacitive load.

We first consider the dynamics of an ideal CMOS inverter without noise as shown in Fig. 3.10. It consists of an nFET and a pFET connected in series, driving

a capacitive output load. We consider only the low power supply case with those transistors operating in the subthreshold region. The drain currents through the nFET and pFET are given by

$$I_n(t) = I_{0n} \frac{W}{L} e^{\frac{\kappa_n V_i(t)}{V_T}} \left(1 - e^{-\frac{V_o(t)}{V_T}} \right) \quad (3.45)$$

$$I_p(t) = I_{0p} \frac{W}{L} e^{\frac{\kappa_p (V_{dd} - V_i(t))}{V_T}} \left(1 - e^{\frac{V_o(t) - V_{dd}}{V_T}} \right) \quad (3.46)$$

where I_{0n} and I_{0p} are the current-scaling parameters for nFET and pFET transistors, $\frac{W}{L}$ is the geometry factor, κ_n and κ_p are the subthreshold slope factors for nFET and pFET. $V_T = \frac{kT}{q}$ is the thermal voltage, where temperature T is assumed constant here. V_i and V_o are the input and output voltages. The differential equation for $V_o(t)$ is given by

$$\frac{dV_o(t)}{dt} = \frac{I_p(t) - I_n(t)}{C} \quad (3.47)$$

A nonlinear differential equation for $V_o(t)$ is obtained by substituting (3.45) and (3.46) into (3.47) and rearranging the terms [114]:

$$\frac{dV_o(t)}{dt} = a(t) + b(t) \sinh \left(\frac{c(t) - V_o(t)}{V_T} \right) \quad (3.48)$$

where

$$a(t) = \frac{2W}{LC} e^{\frac{1}{2} \left[\ln(I_{0p} I_{0n}) + \kappa_p \frac{V_{dd}}{V_T} + (\kappa_n - \kappa_p) \frac{V_i(t)}{V_T} \right]} \times \sinh \left(\frac{1}{2} \left[\ln \frac{I_{0p}}{I_{0n}} + \kappa_p \frac{V_{dd}}{V_T} - (\kappa_p + \kappa_n) \frac{V_i(t)}{V_T} \right] \right) \quad (3.49)$$

$$b(t) = \frac{2W}{LC} e^{\frac{1}{2} \left[\ln(I_{0p} I_{0n}) + (\kappa_p - 1) \frac{V_{dd}}{V_T} + (\kappa_n - \kappa_p) \frac{V_i(t)}{V_T} \right]} \quad (3.50)$$

$$c(t) = \frac{V_T}{2} \ln \frac{I_{0n}}{I_{0p}} + \frac{1 - \kappa_p}{2} V_{dd} + \frac{\kappa_p + \kappa_n}{2} V_i(t) \quad (3.51)$$

We next examine the switching behavior of this digital circuit where the input changes abruptly. We treat it as an ideal step input between 0 and V_{dd} . Input $V_i(t)$

is a constant except at the switching time point. Therefore $a(t)$, $b(t)$, and $c(t)$ are constants after the input transition as well. We can solve the nonlinear differential equation analytically for this special case [114]:

$$V_o(t) = c + V_T \ln \left(\frac{1}{de^{\frac{\sqrt{a^2+b^2}}{V_T}t} - \frac{b}{2\sqrt{a^2+b^2}}} + \frac{a + \sqrt{a^2 + b^2}}{b} \right) \quad (3.52)$$

where $d = \frac{1}{e^{\frac{V_o(0)-c}{V_T} - \frac{a+\sqrt{a^2+b^2}}{b}}}$ and $V_o(0)$ is the initial value of the output voltage. Fig. 3.11 and 3.12 show the inverter output trajectories after input switching for V_{dd} at 0.1 V and 0.01 V. The steady state output voltage is obtained from

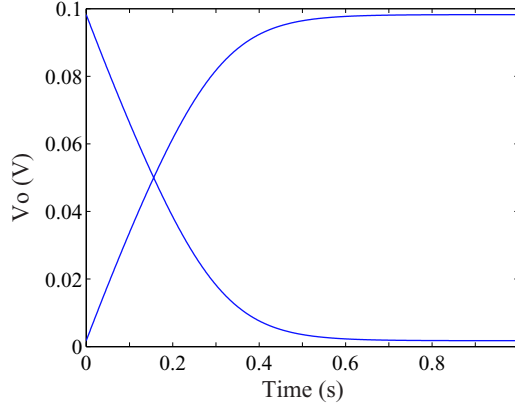


Figure 3.11: Output voltage as a function of time for the ideal CMOS inverter in subthreshold operation after switching at input, $V_{dd} = 0.1$ V.

(3.48) as [114]:

$$V_{ss}(t) = \lim_{t \rightarrow \infty} V_o(t) = c + V_T \ln \left(\frac{a + \sqrt{a^2 + b^2}}{b} \right) \quad (3.53)$$

3.4.3.2 Stochastic Model of a CMOS Inverter

For a CMOS inverter operating at very low power supply, both transistors in the circuit operate in subthreshold or weak inversion. From (3.25) and (3.48) we

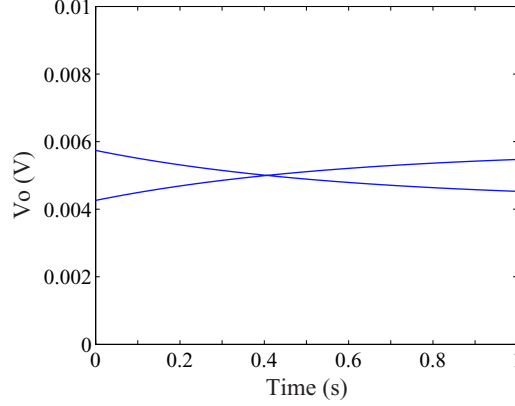


Figure 3.12: Output voltage as a function of time for the ideal CMOS inverter in subthreshold operation after switching at input, $V_{dd} = 0.01$ V.

write the SDE of the inverter output $V_o(t)$ as

$$\begin{aligned}
 dV_o(t) &= \frac{I_p(t) - I_n(t)}{C} dt + \frac{\sqrt{R(t)}}{C} dW \\
 &= \left(a + b \sinh \left(\frac{c - V_o(t)}{V_T} \right) \right) dt + \sqrt{\Gamma(t)} dW(t) \quad (3.54)
 \end{aligned}$$

where $R(t) = q(I_p(t) + I_n(t))$ and $\Gamma(t) = \frac{q(I_p(t) + I_n(t))}{C^2}$.

3.4.3.3 Numerical Simulation of the CMOS Inverter

We apply the Ito integral for numerical solution of the CMOS inverter SDE (3.54) to obtain time domain transient trajectories using Matlab. We use both the Euler and Milstein methods to simulate the switching behavior. We use reasonable parameters from a fabrication process. Fig. 3.13 and 3.14 show the simulated trajectories of inverter outputs from low to high and high to low after inputs switch from 0 to V_{dd} and from V_{dd} to 0 for $V_{dd} = 0.1$ V and 0.01 V. Each simulation generates a unique trajectory. We simulate 100 times and collect 5×10^6 samples from each run long after the switching, assuming that it is already at steady state.

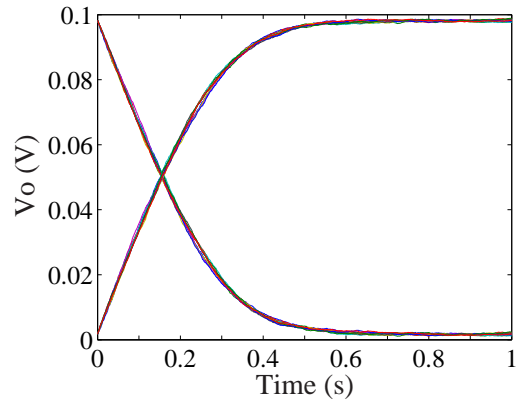


Figure 3.13: Output voltage as a function of time for the CMOS inverter with noise in subthreshold operation after input transition, 10 runs with $V_{dd} = 0.1$ V.

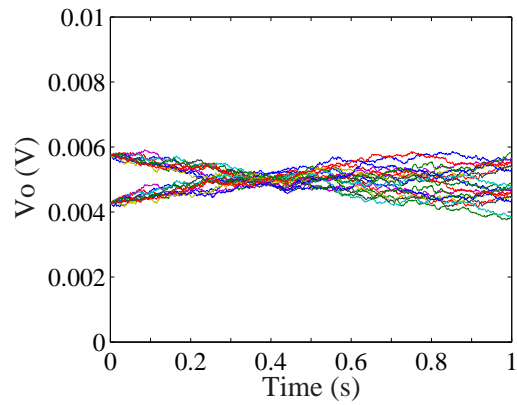


Figure 3.14: Output voltage as a function of time for the CMOS inverter with noise in subthreshold operation after switching at input, 10 runs with $V_{dd} = 0.01$ V.

We compute the probability density function (PDF) of the output voltage at steady state from these samples, as shown in Fig. 3.15. The Euler and Milstein methods give similar results.

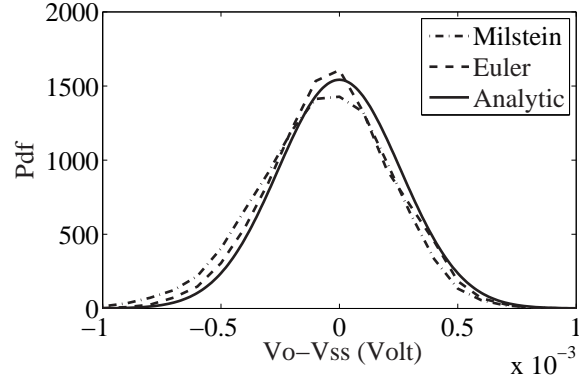


Figure 3.15: The probability density function of output voltage for a CMOS inverter with noise in subthreshold operation, $V_{dd} = 0.1$ V.

3.4.3.4 Analytic Steady State Solution

Since $V_o(t)$ is a stochastic process defined by the SDE, its probability evolution is governed by the Fokker-Planck equation:

$$\frac{\partial P(V, t)}{\partial t} = -\frac{\partial}{\partial V}[A(V)P(V, t)] + \frac{1}{2} \frac{\partial^2}{\partial V^2}[\Gamma(V)P(V, t)] \quad (3.55)$$

where $A(V) = a + b \sinh\left(\frac{c-V}{V_T}\right)$. In digital logic, we are primarily interested in the output voltage once it settles down after switching at the input. Thus we solve for the steady state distribution $P(V, t)$ as $t \rightarrow \infty$, when $\frac{\partial P(V, t)}{\partial t} = 0$. The distribution is centered around the equilibrium value V_{ss} obtained from the noiseless inverter. We approximate $\Gamma(V)$ by its value at $V = V_{ss}$, which is a constant, denoted by Γ for simplicity. The Fokker-Planck equation becomes a second order nonlinear ordinary

differential equation:

$$\frac{1}{2}\Gamma \frac{d^2 P(V)}{dV^2} - A(V) \frac{dP}{dV} + \frac{b}{V_T} \cosh\left(\frac{c-V}{V_T}\right) P = 0 \quad (3.56)$$

Using a first order approximation for cosh and sinh, and introducing a change of variable, we can solve $P(V)$. Assuming deviation of $V_o(t)$ from V_{ss} is small at steady state, we have

$$\cosh\left(\frac{V-V_{ss}}{V_T}\right) \approx 1 \quad (3.57)$$

$$\sinh\left(\frac{V-V_{ss}}{V_T}\right) \approx \frac{V-V_{ss}}{V_T} \quad (3.58)$$

so that

$$\cosh\left(\frac{c-V}{V_T}\right) \approx \cosh\left(\frac{c-V_{ss}}{V_T}\right) - \sinh\left(\frac{c-V_{ss}}{V_T}\right) \frac{V-V_{ss}}{V_T} \quad (3.59)$$

With $a + b \sinh\left(\frac{c-V_{ss}}{V_T}\right) = 0$, we obtain

$$a + b \sinh\left(\frac{c-V}{V_T}\right) \approx -b \frac{V-V_{ss}}{V_T} \cosh\left(\frac{c-V_{ss}}{V_T}\right) \quad (3.60)$$

Substituting them into equation (3.56) and using change of variable $u = \frac{V-V_{ss}}{V_T}$, we have

$$\frac{d^2 P(u)}{du^2} + \alpha u \frac{dP(u)}{du} + (\alpha - \beta u) P(u) = 0 \quad (3.61)$$

where α is defined in (3.65). $\beta = \frac{2bV_T}{\Gamma} \sinh\left(\frac{c-V_{ss}}{V_T}\right) = -\frac{2aV_T}{\Gamma}$. α and β are at the same scale and u will be very small, thus βu can be neglected compared with α . We obtain a simple differential equation which we can solve analytically,

$$\frac{d^2 P(u)}{du^2} + \alpha u \frac{dP(u)}{du} + \alpha P(u) = 0 \quad (3.62)$$

The solution of $P(u)$ is

$$P(u) = \sqrt{\frac{\alpha}{2\pi}} e^{-\frac{\alpha u^2}{2}} \quad (3.63)$$

Therefore,

$$P(V) = \frac{1}{V_T} \sqrt{\frac{\alpha}{2\pi}} e^{-\frac{\alpha}{2} \left(\frac{V-V_{ss}}{V_T}\right)^2} \quad (3.64)$$

$$\alpha = \frac{2bV_T}{\Gamma} \cosh\left(\frac{c-V_{ss}}{V_T}\right) = \frac{2V_T}{\Gamma} \sqrt{a^2 + b^2} \quad (3.65)$$

The $P(V)$ turns out to be a Gaussian distribution centered around V_{ss} . From the solution, we verify the approximations we used in the derivation. For $V_{dd} = 0.1$ V, $\alpha = 10^4$, $\beta = -9.98 \times 10^3$, the standard deviation of the distribution is $\sigma = 2.59 \times 10^{-4}$. Therefore the deviation of $V_o(t)$ from V_{ss} is small at steady state. Within 3σ , $|u| = \left|\frac{V_o(t)-V_{ss}}{V_T}\right| < 0.03$, the approximations in (3.57)-(3.60) are valid. The approximation from (3.61) to (3.62) is valid because $\beta u \ll \alpha$. The solid line in Fig. 3.15 shows the analytical result for $V_{dd} = 0.1$ V, which closely matches the simulation results.

Chapter 4

Fly Inspired Autonomous Navigation: Theory and Simulation

4.1 Background

The tiny fly is a huge success - they have survived over 300 million years and represent one of the most successful animal groups on our planet (i.e. one tenth of the known species is a fly) [48]. This evolutionary success can be largely attributed to their highly efficient sensory and motor systems. Fly photoreceptors are capable of responding to single photons while successfully adapting to intensities up to $\sim 10^6$ effectively absorbed photons per second [47]. Flies can chase mates at turning velocities of more than $3000^\circ/\text{s}$ with delay times of less than 30 ms [48] relying mainly on their visual input with help from other sensory modalities. Microsystems emulating the fly's vision and feedback control systems offer highly efficient solutions for low power, high speed, and robust sensor and actuator systems, therefore relieve the heavy computation and high power consumption required by the conventional frame capture and digital processing based approach. Starting from this chapter, we move up from synapse level to sensorimotor integration level.

4.1.1 The Visual System of the Fly

The visual system of the fly begins with two large compound eyes. Each eye is built from many units called ommatidia. Inside each ommatidium, a lens focuses

light from a very limited visual angle ($\sim 2^\circ$) onto a small group of six to nine photoreceptors, which convert optical signals into electrical signals [116]. The retinal images are processed further by three successive layers of neuropile in the visual ganglia. These layers are the lamina, medulla and lobula complex, in order from peripheral to central. The projection of visual images onto the neuropile is retinotopic, and the spatial relationships among image points are conserved throughout the layers [48].

Large monopolar cells (LMCs) in the lamina effectively encode contrast over the full operating range of the fly's visual system [117]. Cells within the medulla are suggested to encode local motion [118], however their response characteristics are largely unknown due to their small size [48]. The lobula complex comprises two parts: lobula and lobula plate, and retinotopic projections converge at the lobula plate. Visual interneurons called lobula plate tangential cells (LPTCs) have large dendritic trees, integrating visual information across a large field. There are about 60 different LPTCs on each side identified by their characteristic anatomy and response. In general, they are directionally sensitive to visual motion, excited by a preferred direction (PD) and inhibited by a null direction (ND) that opposite to PD. LPTCs can be grouped into horizontally and vertically sensitive cells according to their preferred orientation [48].

There are another two distinct classes of cells: figure detection (FD) and contralateral inhibited (CI). Both respond preferentially to small moving objects, and it is believed that they play a key role in object fixation and pursuit [48].

4.1.2 The Sensorimotor System of the Fly

Flies have superb acrobatic maneuverability. They can hover or maneuver in virtually any direction, maintaining balance within turbulent flow conditions. A typical flight trajectory is a series of straight flight paths interrupted by rapid yaw turns called saccades. The flight musculature of flies is composed of two groups: large power muscles which elevate and depress the wings bilaterally to generate lift for staying aloft; and small steering muscles which alter the path and orientation of each wing stroke [49]. The antagonistic organization of power muscles enables the wings to oscillate up and down at high frequency. At such high frequency, at most single action potential can be sent to drive steering muscles during each wing beat, therefore accurate phasic activation of steering muscles provides subtle modulation of the wing beat [49, 119].

Sensory inputs converge on the thoracic flight motor circuitry. Although a large fraction of these inputs descend from visual processing, a special mechanical sensor, the haltere, has proven to be essential. Independently encoding body rotation along roll, pitch and yaw axes from coriolis forces, halteres operate at high rotational velocities exceeding $800^\circ/s$, while visual response is attenuated beyond $200^\circ/s$. Normally visuo- and mechano-sensory feedback counteracts rotations of the body to maintain straight flight. Occasionally, image expansion caused by approaching objects or sideways slips triggers saccadic turns. Feedback from halteres terminates the rapid turn to limit the rotation to 90° [49, 119]. Fig. 4.1 shows a cartoon of the major sensory feedback paths to the motor system.

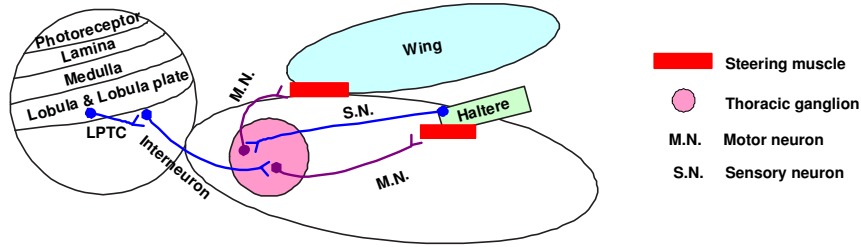


Figure 4.1: Cartoon of fly’s sensory motor systems.

4.1.3 Visual Navigation in Flying Insects

Flying insects have immobile compound eyes with fixed-focus optics, which result in poor spatial resolution and lack of binocular stereopsis. They have developed rather different strategies for visually guided behavior relying mostly on visual cues from image motion. When an insect flies around, moving patterns of illuminance generate local image shifts on the retina, called optic flow. Optic flow over a visual field is called optic flow field or motion field. Hengstenberg, Krapp, and Hengstenberg [120] showed the flying motion in six degrees of freedom (DOF) in a 3-D environment and the corresponding optic flow fields. Optic flow field contains information about self-motion, object motion, and spatial layout of the environment, and plays a critical role in the fly’s vision based navigation. Some important strategies are briefly reviewed here. Thorough reviews and references can be found in [52–54, 56].

- **Flight Stabilization:** When an insect flies along a straight course and is blown to the left by a gust of wind, it compensates for the disturbance by turning to the right. This phenomenon is known as optomotor response and has been extensively investigated in controlled experimental setups [121, 122]. A teth-

ered fly at the center of a drum with a textured wall generates torque towards the same direction as the drum rotates. It is through the optomotor response we know that motion is sensed by cross-correlating the intensity variations in neighboring ommatidia in compound eyes [62, 65].

- **Centering Response:** When bees fly through a tunnel, they tend to stay in the middle of the tunnel [123]. This can help them to negotiate narrow gaps and fly along the safest route. Experiments show that this centering response is accomplished as bees try to balance optic flow from left and right eyes during the flight. If one side of the tunnel is moving along the same direction as the flight path, bees tend to fly closer to the moving side and vice versa. The spatial frequency content and the contrast of the pattern on the sides does not affect the behavior, suggesting that bees have the capability to extract the actual speed of the image independently of the patterns [123].
- **Object Avoidance and Landing:** When an insect approaches an object, the image of this object expands. Insects use this optic flow expansion to detect obstacles or surfaces. They either turn away from the imminent collision or initiate smooth landing. Although similar motion cues are used, these two different responses are mediated by separate pathways in the fruit fly [124]. For landing, optic flow is also used to reduce the flight speed accurately for touchdown [125].
- **Odometry:** Bees have demonstrated the ability to effectively encode the distance to a destination and communicate it to others so that they can fly to

the same location. Experiments show that the distance estimation is strongly related to image motion. For a bee flying through a tunnel, sidewall motion in the same direction as the flight causes the bee to believe it has flown a shorter distance [126]. Vision-based odometry might be more robust than energy-based approaches because it won't be as sensitive to wind and changing loads that bees might have to carry (e.g. pollen).

- **Chasing Behavior and Motion Camouflage:** For individual and species survival, insects need to chase their potential mates, capture their prey, and escape from their enemies. All of these rely on their ability to see targets, often small and fast moving, to track them in the visual field, and to react. Special neurons that respond selectively to small moving objects have been identified [127]. Control strategies, such as keeping the target in the center of the visual view [128] or maintaining a fixed angle subtended by the target [129] have been found in the behavior of some flies. It has also been observed that some insects appear to conceal themselves by moving in a way that makes them appear stationary to their targets [130]. Justh and Krishnaprasad [131] have derived a control law for this "motion camouflage".

4.1.4 Natural Environment

Simple stimuli have been widely used in biological experiments because they are easy to produce and control. Only recently has it become technically feasible to introduce complex stimuli close to the natural visual environment where animals

typically operate. The neuronal responses to complex stimuli are found to be quite different from the responses to simple stimuli. The response of motion-sensitive neurons to simple stimuli is ambiguous, but natural signals allow the fly to disambiguate the interpretation [132, 133]. This is not surprising, as nervous systems have evolved to compute behaviorally relevant information by taking advantage of natural stimuli statistics [134] and the animals' own unique behaviors [135, 136].

Insect neural systems are relatively simple, and it is feasible to emulate many of their structures and functions using current technology. Not only does this enable biologically-inspired robots with high performance and efficiency, it also allows us to test our understanding of the biological system. By building the system and testing it in its natural environment, we expect to encounter and resolve critical and practical issues that are never seen in a perfect simulation environment [70].

4.2 Optic Flow Wide Field Integration (WFI) based Navigation

Recent experimental results revealed that there is fine spatial structure of the motion direction sensitivity in the receptive field of LPTCs beyond their preferred global direction [57]. It has been postulated that such matched filter structures correspond to neuronal models of the external world [58]. LPTCs may compute specific self-motion parameters by filtering its vast visual input stream through synaptic connections. In total, 60 LPTCs on each side extract rich information about the motion and the environment, which provide feedback to the motor systems for flight control and navigation.

4.2.1 WFI based Navigation

It has been shown that motion cues for navigation can be extracted via WFI of the retinal optic flow over the full 2π view angle around the center of a circular sensor [2,59–61]. WFI is computed by circular Fourier decomposition of the optic flow with the orthonormal function basis $\Phi = \{\frac{1}{\sqrt{2}}\} \cup \{\cos n\gamma : n = 1, 2, \dots\} \cup \{\sin n\gamma : n = 1, 2, \dots\}$. Each basis function is used as a filter $F_i(\gamma)$, and the WFI is performed as the inner product between the optic flow field and the filter, $\frac{1}{2\pi} \int_{-\pi}^{\pi} \dot{Q}(\gamma) \cdot F_i(\gamma) d\gamma$, where γ is the direction of the view. The WFI outputs have direct relation with the lateral and rotational dynamics and forward speed of the vehicle. With appropriate gain, the motion cue can be used in a feedback control loop to drive the vehicle towards a desired state or trajectory [2,59–61]. It is also demonstrated that this output feedback methodology is sufficient to explain some of the experimentally observed honeybee flight behavior [60]. Fig. 4.2 shows a framework of using optic flow WFI in feedback control for navigation.

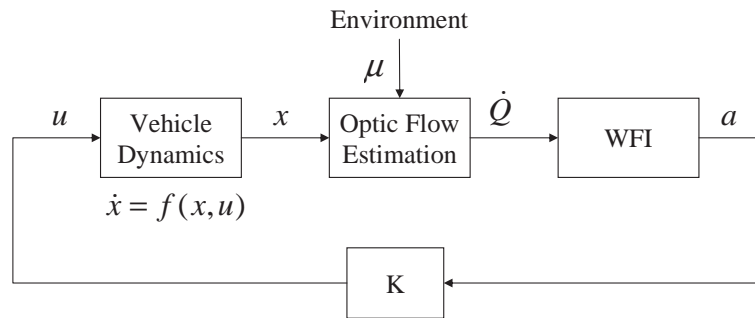


Figure 4.2: Optic flow WFI based feedback control [2].

4.2.2 Elementary Motion Detector

The Reichardt correlation model was first proposed by Reichardt and his colleagues as the elementary motion detector (EMD) in flies [62–65]. Additional experimental evidence has accumulated and this model has been extended to invertebrate visual motion as well. Mathematically, it is equivalent to the spatiotemporal energy model for motion detection in vertebrate motion vision [137]. The basic unit of a

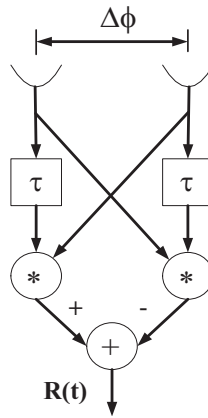


Figure 4.3: Reichardt EMD.

Reichardt EMD (REMD) is a correlator between a delayed input and its neighboring input as shown in Fig. 4.3. Each REMD consists of two such units sensitive to opposite directions, and the differential output imparts selectivity to motion along the orientational direction between inputs. Two REMDs aligned perpendicularly give 2-D motion information subject to the aperture problem. A low-pass filter with time constant τ is often used as the delay element. The REMD response to a sinusoidal grating with amplitude ΔI moving to the right at velocity v is [138]:

$$R(t) = \Delta I^2 \frac{2\pi\tau v/\lambda}{1 + (2\pi\tau v/\lambda)^2} \cdot \sin(2\pi\Delta\phi/\lambda), \quad (4.1)$$

where $f_t(\text{cyc}/s)$ and $f_s(\text{cyc}/^\circ)$ are temporal and spatial frequency, λ is the wavelength $1/f_s$, and $v = f_t/f_s$. Clearly the REMD response is not just a function of velocity, but depends on contrast and compound effect of f_t and f_s .

When the distance from an object to the fly changes, the apparent motion also changes. The closer the object is, the bigger the apparent motion is. Apparent angular velocity $v = f_t/f_s$, the change of v could be caused by change in either f_t or f_s . Natural image statistics are found to be invariant to scale [139, 140]. This implies that the spatial power spectrum stays same whether the image is taken closer or farther: more details are seen at smaller distance, but the span of view angle of the detail increases as well because of smaller distance. The optics of fly's eyes also limit the spatial frequency to a quite narrow band ($0.25 \text{ cyc}/^\circ$). With fixed spatial frequency content, image velocity is proportional to the temporal frequency. Therefore velocity tuning is mainly dependent on temporal frequency tuning.

Nonzero DC intensity in the pattern creates oscillations around $R(t)$, but these variations can mostly be cancelled by spatial averaging. Various elaborations of the basic REMD have been proposed to match more closely to physiological results and to obtain better velocity constancy, i.e., velocity estimation independent of environment [136, 141, 142]. Fig. 4.4 shows one elaborated EMD model. A velocity estimator has been built using nonlinear summation of signals from an EMD array [143]. Various components have been evaluated for the purpose of velocity constancy, such as spatial filter, temporal filter, nonlinear saturation, nonlinear integration, and motion adaptation. With the spatial frequency content of natural scenes invariant in space, the REMD response (4.1) shows that another important factor for velocity

estimation is contrast change (contrast is defined as $\Delta I/I$). Contrast adaptation has been proposed in EMD to reduce the its sensitivity to contrast change. [143]. The bell shape of the temporal frequency tuning curve also causes the ambiguity of velocity estimation.

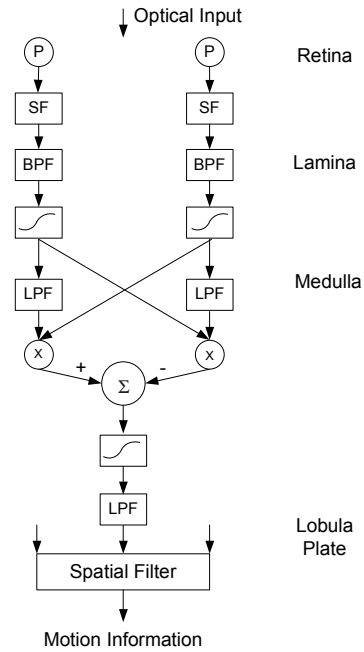


Figure 4.4: Elaborated EMD. P: photoreceptor, SF: spatial filter, BPF: band-pass filter, LPF: low-pass filter.

4.2.3 EMD in WFI based Navigation

In the Simulink model developed before [59], a vehicle moves on a horizontal plane with three DOF (planar translation with single-axis rotation). The environment is defined by a bitmap. Ideal optic flow field is computed from the sensor geometry, the vehicle position and movement, and the environment. Force and torque control inputs are generated from the ideal optic flow field by the spatial

matched filters. Here we evaluate REMD and its variation as the optic flow estimator in WFI based navigation.

We first investigated the effect of the nonlinear response of the REMD on the navigation performance. The ideal optic flow is transformed by the velocity tuning curve of the REMD (4.1) for fixed λ before the spatial filtering. The vehicle can successfully navigate through the environment as long as the velocity is in the monotonic range of the tuning curve. It shows importance of the monotonicity of the response in WFI-based navigation because the algorithm relies on the relative spatial response to the motion.

Motion sensitive neurons with different intrinsic time constants (slow and fast optimal velocity), are found in insects [144]. Multiple motion detectors with different spatiotemporal tuning curves might collectively help insects to estimate the image velocity independent of the environment. In theory, the normalized responses of two EMDs with different time constants show no dependence on contrast and have a much wider range of response monotonicity:

$$\frac{R_2}{R_1 + R_2} = \frac{\tau_2}{\tau_1 + \tau_2} \cdot \frac{1 + \tau_1^2 \omega^2}{1 + \tau_1 \tau_2 \omega^2} \quad (4.2)$$

where R_1 and R_2 are responses of REMDs with time constants τ_1 and τ_2 respectively. Fig. 4.5 shows the velocity tuning curve R_1 with peak response at $10^\circ/s$ and R_2 with peak response at $50^\circ/s$ and the normalized response $R = \frac{R_2}{|R_1 + R_2| + c}$, a small positive constant c is added to avoid the division by zero at zero velocity. Divisive normalization is also used in modeling nonlinearities and adaptation in cortical neurons [145]. Normalized response from two velocity tuning curves with peak

velocity $20^\circ/s$ and $200^\circ/s$ is used in the simulation and the performance is improved because of the extended range of the monotonic response.

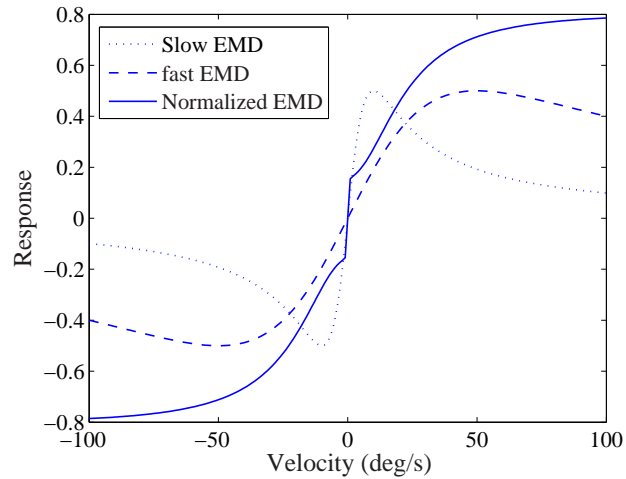


Figure 4.5: Velocity tuning from slow EMD (peak velocity = $10^\circ/s$), fast EMD (peak velocity = $50^\circ/s$), and normalized fast EMD.

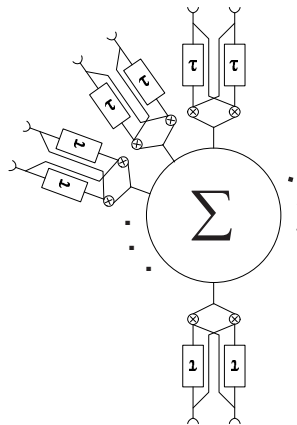


Figure 4.6: EMD circular sensor.

To further test the EMD, I introduced a circular motion sensor model with elaborated EMDs evenly distributed over 2π view angle, as shown in Fig 4.6. The image projected to each EMD is computed according to its position, orientation, and

the environment. When the vehicle is moving, EMDs compute the local optic flow resulting from their changing inputs. WFI of the local optic flow provides feedback to control the force and torque of the vehicle.

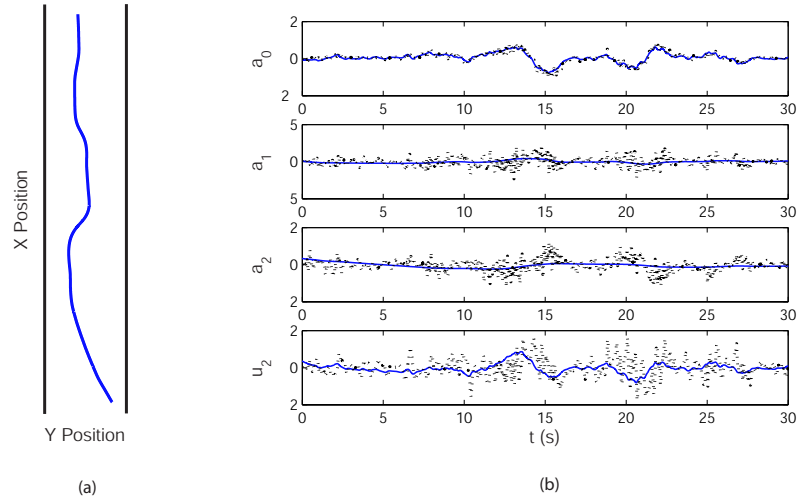


Figure 4.7: Simulation of WFI based navigation with EMD sensors. (a) Navigation through a tunnel, (b) Four matched filter outputs from EMD based optic flow (dotted line) and ideal optic flow (solid line), a_0 : DC filter, a_1 : $\cos(\cdot)$ filter, a_2 : $\cos(2\cdot)$ filter, u_2 : torque filter.

Due to the limited capability to simulate realistic environments in Simulink, the environment is simulated as a parallel tunnel with natural images painted on the tunnel wall. Spatial filtering or averaging of the image projected onto the EMD is critical because it emulates the effect of scaling as the vehicle moves close to or away from the wall. Temporal pre-filtering with low-pass and high-pass filters, nonlinear saturation, and proper monotonic velocity response range are implemented. The output of a single EMD is very noisy because of the transient nature of its response and it poses quite a challenge in making the closed-loop simulation work. Therefore,

at each azimuthal position, 5 EMDs along the vertical direction are used and their outputs are averaged before filtering. This improves the performance. Fig. 4.7 shows one simulation result. The WFI outputs from EMD-based optic flow fluctuate around the WFI outputs from the ideal optic flow.

This simulation demonstrates the possibility of using EMDs in optic flow WFI based navigation and identifies important factors. In any actual system, there will be more noise and non-ideality. Sensors of other modalities might be needed for flight stabilization so that visual information can be used just for navigation. Other matched filters might be needed for enhanced capability, such as an optic flow expansion filter for imminent obstacle avoidance.

4.3 Limited View Angle Solution

WFI based navigation has been formulated and demonstrated using optic flow of a 2π view angle from a circular sensor [2, 60, 61]. Limited view angles much less than 2π seem not prevent flying insects from navigating around well. It is also reasonable to assume that the most relevant visual information for flight control would come from the front of the animal. Therefore we extend previous work to motion cues recovery and feedback control from a limited view angle. This also alleviates technical difficulties associated with constructing a full circular motion sensor. We follow the same methodology developed by Humbert, Murray, and Dickinson [2, 60]. We first demonstrate the case with a view angle of π , then generalize it to any view angles less than π .

4.3.1 View Angle of π

Fig. 4.8 shows the configuration of a ground vehicle in a tunnel environment [2, 60]. The coordinate $\{x, y\}$ sits in the middle of the tunnel, providing the inertial frame, where a is the half width of the tunnel. $\{x_b, y_b\}$ is the body frame attached to the body of the vehicle. θ is the rotation angle of the body frame referenced to the inertial frame. γ is the angle of a point in the environment and $r(\gamma)$ is the corresponding position vector in the body frame.

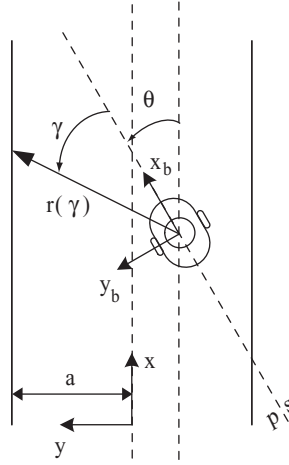


Figure 4.8: Vehicle configuration in planar tunnel.

For a planar motion, the instantaneous optic flow on a circular sensor is a 2π periodic function of the azimuth angular γ relative to the body frame [2, 60]:

$$\dot{Q} = -\dot{\theta} + \mu(\gamma)(\dot{x}_b \sin \gamma - \dot{y}_b \cos \gamma), \quad (4.3)$$

where $\dot{\theta}$, \dot{x}_b , \dot{y}_b are the kinematics of the body frame, and $\mu(\gamma) = 1/r(\gamma)$ is the nearness function along direction γ . With a view angle of π , γ is in $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Therefore the WFI or the spatial filtering of optic flow is performed within $[-\frac{\pi}{2}, \frac{\pi}{2}]$ using a

function set $\Phi = \{1\} \cup \{\cos n\gamma : n = 1, 2, \dots\} \cup \{\sin n\gamma : n = 1, 2, \dots\}$:

$$a_0(t) = \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \dot{Q}(\gamma) d\gamma \quad (4.4)$$

$$a_n(t) = \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \dot{Q}(\gamma) \cdot \cos n\gamma d\gamma \quad (4.5)$$

$$b_n(t) = \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \dot{Q}(\gamma) \cdot \sin n\gamma d\gamma \quad (4.6)$$

Applying the same spatial filters on the nearness function $\mu(\gamma)$:

$$A_0(t) = \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \mu(\gamma) d\gamma \quad (4.7)$$

$$A_n(t) = \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \mu(\gamma) \cdot \cos n\gamma d\gamma \quad (4.8)$$

$$B_n(t) = \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \mu(\gamma) \cdot \sin n\gamma d\gamma \quad (4.9)$$

we can rewrite the WFI of optic flow in terms of the vehicle kinematics $(\dot{x}_b, \dot{y}_b, \dot{\theta})$

and $\{A_0, A_k, B_k : k = 1, 2, \dots\}$:

$$a_0 = -\dot{\theta} + \dot{x}_b B_1 - \dot{y}_b A_1 \quad (4.10)$$

$$a_{2n} = \frac{\dot{x}_b}{2} (B_{2n+1} - B_{2n-1}) - \frac{\dot{y}_b}{2} (A_{2n+1} + A_{2n-1}) \quad (4.11)$$

$$b_{2n} = \frac{\dot{x}_b}{2} (A_{2n-1} - A_{2n+1}) - \frac{\dot{y}_b}{2} (B_{2n+1} + B_{2n-1}) \quad (4.12)$$

$$a_{2n+1} = -\frac{(-1)^n}{(2n+1)} \frac{2\dot{\theta}}{\pi} + \frac{\dot{x}_b}{2} (B_{2n+2} - B_{2n}) - \frac{\dot{y}_b}{2} (A_{2n+2} + A_{2n}) \quad (4.13)$$

$$b_{2n+1} = \frac{\dot{x}_b}{2} (A_{2n} - A_{2n+2}) - \frac{\dot{y}_b}{2} (B_{2n+2} + B_{2n}) \quad (4.14)$$

For the planar tunnel geometry, the nearness function $\mu(\gamma)$ can be expressed as a function of the lateral position y , body frame orientation θ , and the tunnel half-width a [2, 60], defined for $\gamma \in [-\frac{\pi}{2}, \frac{\pi}{2}]$:

$$\mu(\gamma) = \begin{cases} \frac{\sin(\gamma+\theta)}{a-y} & -\theta \leq \gamma \leq \frac{\pi}{2} \\ -\frac{\sin(\gamma+\theta)}{a+y} & -\frac{\pi}{2} \leq \gamma \leq -\theta \end{cases} \quad (4.15)$$

The deviation of the body frame orientation from the forward direction of the tunnel is always less than or equal to $\frac{\pi}{2}$, so $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. We compute the spatial filter outputs of this nearness function:

$$A_0 = \frac{1}{\pi} \cdot \frac{2a + 2y \sin \theta}{a^2 - y^2} \quad (4.16)$$

$$A_1 = \frac{2a \cos \theta + (\pi y + 2a\theta) \sin \theta}{2\pi(a^2 - y^2)} \quad (4.17)$$

$$B_1 = \frac{\cos \theta(\pi y + 2a\theta)}{2\pi(a^2 - y^2)} \quad (4.18)$$

$$A_k = \begin{cases} \frac{2}{\pi} \frac{-a \cos k\theta + (-1)^{m+1} y \sin \theta}{(k^2 - 1)(a^2 - y^2)} & k = 2m \\ \frac{2}{\pi} \frac{-a \cos k\theta + (-1)^m a k \cos \theta}{(k^2 - 1)(a^2 - y^2)} & k = 2m + 1 \end{cases} \quad (m > 0) \quad (4.19)$$

$$B_k = \begin{cases} \frac{2}{\pi} \frac{a \sin k\theta + (-1)^{m+1} k y \cos \theta}{(k^2 - 1)(a^2 - y^2)} & k = 2m \\ \frac{2}{\pi} \frac{a \sin k\theta + (-1)^{m+1} a \sin \theta}{(k^2 - 1)(a^2 - y^2)} & k = 2m + 1 \end{cases} \quad (m > 0) \quad (4.20)$$

Furthermore by changing the coordinate from body frame to inertial coordinate using

$$\dot{x}_b = \dot{x} \cos \theta + \dot{y} \sin \theta \quad (4.21)$$

$$\dot{y}_b = -\dot{x} \sin \theta + \dot{y} \cos \theta, \quad (4.22)$$

we can write the transformation of optic flow in inertial coordinates [2, 60]. The first

several outputs are shown here:

$$a_0 = -\dot{\theta} + \frac{\dot{x}(\pi y + 2a\theta + a \sin 2\theta) - 2a\dot{y} \cos^2 \theta}{2\pi(a^2 - y^2)} \quad (4.23)$$

$$a_1 = -\frac{2}{\pi}\dot{\theta} - \frac{-4a\dot{x} \sin \theta + \dot{x}y \cos 2\theta - 3\dot{x}y + 2a\dot{y} \cos \theta + y\dot{y} \sin 2\theta}{3\pi(a^2 - y^2)} \quad (4.24)$$

$$a_2 = -\frac{-a\dot{x} \sin 2\theta + 2a\theta\dot{x} \cos 2\theta + \pi\dot{x}y \cos 2\theta + 2a\theta\dot{y} \sin 2\theta + \pi\dot{y}y \sin 2\theta}{4\pi(a^2 - y^2)} \quad (4.25)$$

$$a_3 = \frac{2}{3\pi}\dot{\theta} - \frac{4a\dot{x} \sin 3\theta + 9\dot{x}y \cos 2\theta + 5\dot{x}y - 6a\dot{y} \cos 3\theta + 9\dot{y}y \sin 2\theta}{15\pi(a^2 - y^2)} \quad (4.26)$$

$$b_1 = -\frac{2(-2a\dot{x} \cos \theta - \dot{x}y \sin 2\theta - a\dot{y} \sin \theta + \dot{y}y \cos 2\theta)}{3\pi(a^2 - y^2)} \quad (4.27)$$

$$b_2 = -\frac{-2a\dot{x} \cos 2\theta - 2a\dot{x} - 2a\theta\dot{x} \sin 2\theta - \pi\dot{x}y \sin 2\theta}{4\pi(a^2 - y^2)} + \frac{-a\dot{y} \sin 2\theta + 2a\theta\dot{y} \cos 2\theta + \pi\dot{y}y \cos 2\theta}{4\pi(a^2 - y^2)} \quad (4.28)$$

$$b_3 = -\frac{2(2a\dot{x} \cos 3\theta - 3\dot{x}y \sin 2\theta + 3a\dot{y} \sin 3\theta + 3\dot{y}y \cos 2\theta)}{15\pi(a^2 - y^2)} \quad (4.29)$$

We consider rolling or wheeled vehicles of the unicycle type, subject to the nonholonomic constraint $\dot{x} \sin \theta - \dot{y} \cos \theta = 0$, which enforces $\dot{y}_b = 0$. Similar results are obtained for a hovercraft with three DOF as well. The kinematic and dynamic of the motion in the inertial configuration are [2, 60]

$$\dot{x} = v \cos \theta \quad (4.30)$$

$$\dot{y} = v \sin \theta \quad (4.31)$$

$$m\dot{v} = (T_s + T_p)/r_w \quad (4.32)$$

$$J\ddot{\theta} = r_0(T_s - T_p)/r_w \quad (4.33)$$

where $v = \dot{x}_b$, T_s and T_p are starboard and port wheel torques, r_0 and r_w are vehicle half width and wheel radius. Assuming small states (other than v) and control inputs, the linearized equations of the state variable $p = (v, y, \theta, \dot{\theta})^T$ near a

centerline flight trajectory $p_0 = (v_0, 0, 0, 0)$ are

$$\dot{p} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & v_0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} p + \begin{pmatrix} u_1/m \\ 0 \\ 0 \\ u_2/J \end{pmatrix} \quad (4.34)$$

where $u_1 = (T_s + T_p)/r_w$, $u_2 = r_0(T_s - T_p)/r_w$.

We linearize a_i and b_i with respect to the state variable $p = (v, y, \theta, \dot{\theta})^T$ around $p_0 = (v_0, 0, 0, 0)^T$ so that $z(p) = z(p_0) + (p - p_0)^T \frac{\partial z}{\partial p} |_{p_0}$:

$$\begin{pmatrix} z_{a_0} \\ z_{a'_1} \\ z_{a_2} \\ z_{b_1} \end{pmatrix} = \begin{pmatrix} 0 & \frac{v_0}{2a^2} & \frac{v_0}{\pi a} & -1 \\ 0 & -\frac{8v_0}{3\pi a^2} & 0 & -\frac{8}{3\pi} \\ 0 & -\frac{v_0}{4a^2} & 0 & 0 \\ \frac{4}{3\pi a} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} v \\ y \\ \theta \\ \dot{\theta} \end{pmatrix} \quad (4.35)$$

where $z_{a'_1} = 3z_{a_1} + 5z_{a_3}$ to simplify its dependance to the state variables. The matrix has a full rank. (4.34) shows that the state variable v is decoupled from the rest of the state variables, and (4.35) shows that z_{b_1} is only dependent on v . Therefore the feedback control can be broken into forward speed regulation and centering regulation, similar to the 2π view angle case [2, 60].

If we choose $u_1 = K_{11}(\frac{4v_0}{3\pi a} - z_{b_1})$, the linearized closed loop dynamics of v becomes

$$\dot{v} = -\frac{1}{m} \cdot \frac{4}{3\pi a} K_{11}(v - v_0) \quad (4.36)$$

The local stability of the nonlinear system is achieved when $K_{11} > 0$.

If we choose $u_2 = K_{20}z_{a_0} + K_{21}z_{a'_1} + K_{22}z_{a_2}$, the characteristic equation for the

linearized closed loop dynamics of $(y, \theta, \dot{\theta})$ is

$$s^3 + \frac{1}{J}(K_{20} + \frac{8}{3\pi}K_{21})s^2 - \frac{v_0}{J\pi a}K_{20}s + \frac{v_0^2}{12a^2J}(-6K_{20} + \frac{32}{\pi}K_{21} + 3K_{22}) = 0 \quad (4.37)$$

To achieve a stable response, we require $K_{20} < 0$, $K_{21} > -\frac{3\pi}{8}K_{20}$, $K_{22} > 2K_{20} - \frac{32}{3\pi}K_{21}$, and $(K_{20} + \frac{8}{3\pi}K_{21})(-\frac{1}{J\pi}K_{20}) > \frac{v_0}{12a}(-6K_{20} + \frac{32}{\pi}K_{21} + 3K_{22})$.

We applied the same method to the view angle $\pi/2$. The linearized filtered outputs are

$$\begin{pmatrix} z_{a_0} \\ z_{a_1} \\ z_{a_2} \\ z_{b_2} \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{4} \frac{-2v_0 + v_0\pi}{a^2} & \frac{1}{2} \frac{v_0}{a} & -\frac{1}{2}\pi \\ 0 & \frac{1}{6} \frac{v_0\sqrt{2}}{a^2} & \frac{1}{6} \frac{-\sqrt{2}v_0 + 4v_0}{a} & -\sqrt{2} \\ 0 & -\frac{1}{16} \frac{-8v_0 + 2v_0\pi}{a^2} & \frac{1}{4} \frac{v_0}{a} & -1 \\ \frac{1}{4} \frac{1}{a} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} v \\ y \\ \theta \\ \dot{\theta} \end{pmatrix} \quad (4.38)$$

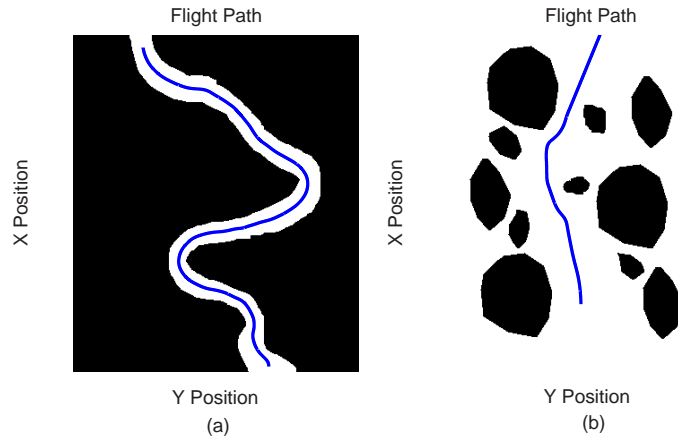


Figure 4.9: Simulation of WFI based navigation using limited view angle optic flow.

a)View angle: π , b)View angle: $\pi/2$.

The matrix has a full rank, indicating the system is controllable with full feedback. Using the criterion of stability to choose the feedback gain factor K_{11} ,

K_{20} , K_{21} , and K_{22} , we simulated the navigation of the wheeled vehicle in the tunnel environment with the view angle of π and $\pi/2$. To obtain good navigation behavior, once we chose the gain factors according to the Routh-Hurwitz rule, the eigenvalues of the matrix for the system with feedback are computed. Adjustment is made to avoid eigenvalue on the imaginary axis. In both cases, the vehicle navigated the test environment successfully. Fig. 4.9 shows the simulated navigation path through the test environment in each of these cases.

4.3.2 General Case

Now we generalize the result to WFI on $[-\beta, \beta]$, $0 < \beta \leq \frac{\pi}{2}$ for wheeled vehicles with the nonholonomic constraint. The integration of optic flow is performed across $[-\beta, \beta]$ using the same function set $\Phi = \{1\} \cup \{\cos n\gamma : n = 1, 2, \dots\} \cup \{\sin n\gamma : n = 1, 2, \dots\}$ as before. The first four linearized outputs around the centerline are:

$$\begin{pmatrix} z_{a_0} \\ z_{a_1} \\ z_{a_2} \\ z_{b_1} \end{pmatrix} = \begin{pmatrix} 0 & Z_{3 \times 3} \\ \alpha & 0 \end{pmatrix} \begin{pmatrix} v \\ y \\ \theta \\ \dot{\theta} \end{pmatrix} \quad (4.39)$$

where $\alpha = \frac{1}{6a}(8 + \cos(3\beta) - 9 \cos(\beta))$ and

$$Z_{3 \times 3} = \begin{pmatrix} \frac{v_0}{2a^2}(2\beta - \sin 2\beta) & \frac{v_0}{2a}(1 - \cos 2\beta) & -2\beta \\ \frac{-v_0}{6a^2}(-3 \sin \beta + \sin 3\beta) & -\frac{v_0}{6a}(\cos 3\beta + 3 \cos \beta - 4) & -2 \sin \beta \\ \frac{v_0}{8a^2}(-\sin 4\beta + 4 \sin 2\beta - 4\beta) & \frac{v_0}{8a}(1 - \cos 4\beta) & -\sin 2\beta \end{pmatrix} \quad (4.40)$$

The rank of $Z_{3 \times 3}$ is 3.

For a linear system with an n -dimension state vector x and an l -dimension observation vector y described by the state differential equation:

$$\dot{x} = Ax + Bu \quad (4.41)$$

$$y = Cx \quad (4.42)$$

where A is an $n \times n$ matrix, B is an $n \times m$ matrix, C is an $l \times n$ matrix, and u is an m -dimension input vector, the system is controllable if the rank of controllability matrix P_c is n [146],

$$P_c = \begin{pmatrix} B & AB & \dots & A^{n-1}B \end{pmatrix} \quad (4.43)$$

The system is observable if the output has a component due to each state variable, meaning that the rank of the observability matrix P_o is n [146],

$$P_o = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix} \quad (4.44)$$

For the wheeled vehicle dynamics (4.34),

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & v_0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad (4.45)$$

$$B = I \quad (4.46)$$

$$C = \begin{pmatrix} 0 & Z_{3 \times 3} \\ \alpha & 0 \end{pmatrix} \quad (4.47)$$

Both the controllability matrix P_c and the observability matrix P_o have rank 4, so the system is controllable and observable.

In both matrix A and C , v and $(y, \theta, \dot{\theta})^T$ are decoupled, thus the control can be broken into the control of v and the control of $q = (y, \theta, \dot{\theta})^T$. If we choose $u_1 = K_3(\alpha v_0 - z_{b1})$, $\dot{v} = -\frac{1}{m} \cdot \alpha K_3(v - v_0)$ with $\alpha > 0$, local stability is achieved when $K_3 > 0$. If we choose $u_2 = K_0 z_{a0} + K_1 z_{a1} + K_2 z_{a2}$,

$$u_2 = K Z_{3 \times 3} \begin{pmatrix} y \\ \theta \\ \dot{\theta} \end{pmatrix} = K \begin{pmatrix} Z_{.1} & Z_{.2} & Z_{.3} \end{pmatrix} \begin{pmatrix} y \\ \theta \\ \dot{\theta} \end{pmatrix} \quad (4.48)$$

where $K = (K_0, K_1, K_2)$, $Z_{.i}$ is the i th column vector of $Z_{3 \times 3}$ for $i = 1, 2, 3$.

The linearized system equation of q with the feedback output is

$$\dot{q} = \begin{pmatrix} 0 & v_0 & 0 \\ 0 & 0 & 1 \\ \frac{r_0}{J} K Z_{.1} & \frac{r_0}{J} K Z_{.2} & \frac{r_0}{J} K Z_{.3} \end{pmatrix} q \quad (4.49)$$

The characteristic equation for the linearized closed loop dynamics is:

$$s^3 - \frac{r_0}{J} K Z_{.3} s^2 - \frac{r_0}{J} K Z_{.2} s - \frac{r_0 v_0}{J} K Z_{.1} = 0 \quad (4.50)$$

To achieve a stable response, all the roots of the characteristic equation should lie in the left half of the s-plane. Routh-Hurwitz criterion gives a necessary and sufficient criterion for the stability of linear systems [146]. Specifically, for a third-order system with characteristic equation of $q(s) = a_3 s^3 + a_2 s^2 + a_1 s + a_0$ to be stable, it is necessary and sufficient that $a_0, a_1, a_2, a_3 > 0$, and $a_2 a_1 \geq a_0 a_3$. Therefore we

require the following three conditions:

$$-\frac{r_0}{J}KZ_{.3} > 0 \quad (4.51)$$

$$-\frac{r_0v_0}{J}KZ_{.1} > 0 \quad (4.52)$$

$$\frac{r_0}{J}KZ_{.3} \cdot \frac{r_0}{J}KZ_{.2} > -\frac{r_0v_0}{J}KZ_{.1} \quad (4.53)$$

Since $Z_{3 \times 3}$ is a matrix with full rank, we can always find such K so that $KZ_{3 \times 3} < 0$. The last condition can be satisfied if we scale K with a large enough positive number. With the condition satisfied, the linearized system is strictly stable, thus the equilibrium point is asymptotically stable for the nonlinear system, according to the Lyapunov's linearization theorem [147].

Chapter 5

Motion Image Sensor with On-chip Adaptation and Programmable Filtering

To achieve a single chip solution to the wide field integration (WFI) based navigation, both optic flow computation and spatial filtering need to be implemented on chip. In previous work [66, 67, 69, 148, 149] where elementary motion detector (EMD) was used for navigation, EMD responses across the field of view were spatially averaged to provide the velocity information about the motion. And it is the only parameter extracted and used. In this case, individual differences among EMDs due to mismatch is not critical. While for the WFI based navigation, detailed spatial structure of the optic flow field is used to extract multiple dynamic and kinematic parameters, therefore the intrinsic difference among EMDs due to mismatch would distort the original optic flow spatial pattern and pose a serious problem. We propose a novel approach to mismatch compensation in the EMDs. We further demonstrate how the same structure can be used to program the analog spatial filters and perform filtering operation. This method can also be applied to other sensory front-end in integrated sensors where on-chip spatial processing is required and distortion from fabrication mismatch among sensor units has to be reduced. The sensor is demonstrated with a ground robot for autonomous navigation.

5.1 EMD Implementation and Mismatch

To reduce the circuit complexity thus to reduce the mismatch, we design the EMD as simple as possible. We use voltage-mode EMD with differential current output so that further operation can be performed in current mode. This implementation is similar to Harrison's implementation in the analog IC for visual collision detection [69].

5.1.1 EMD Implementation

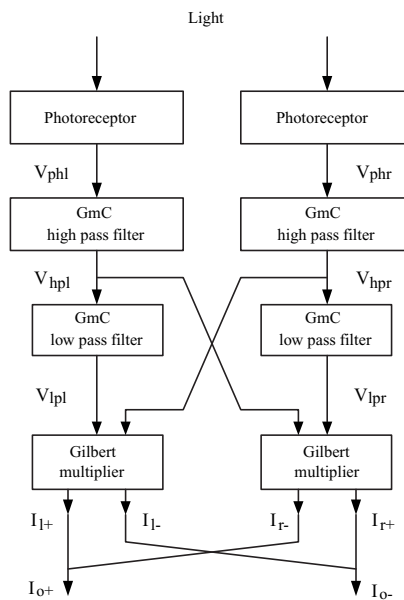


Figure 5.1: Block diagram of EMD implementation.

Fig. 5.1 shows the block diagram of the EMD circuit. Light is absorbed by a pair of photoreceptors and converted to a pair of voltages V_{phl} at left and V_{phr} at right. Both voltage signals pass through the GmC type high-pass filters to remove DC component. Each high-pass filtered signal passes through a GmC type low-pass

filter to obtain a delayed version of the signal. At each branch of the EMD, the delayed signal and the non-delayed signal from the opposite side are multiplied by the Gilbert multiplier. Thus we obtain differential current outputs I_{l+} and I_{l-} from V_{lpl} and V_{hpr} at the left branch, and differential current output I_{r+} and I_{r-} from V_{lpr} and V_{hpl} at the right branch. The EMD output is given by the difference between the left branch $I_{l+} - I_{l-}$ and the right branch $I_{r+} - I_{r-}$, thus we combine current I_{l+} and I_{r-} into I_{o+} , and I_{r+} and I_{l-} into I_{o-} . The output of the EMD I_o is

$$I_o = I_{o+} - I_{o-} = (I_{l+} + I_{r-}) - (I_{r+} + I_{l-}) = (I_{l+} - I_{l-}) - (I_{r+} - I_{r-}) \quad (5.1)$$

For this configuration, $I_o > 0$ for the movement from the left photoreceptor to the right one, and $I_o < 0$ for the opposite direction.

5.1.1.1 Photoreceptor

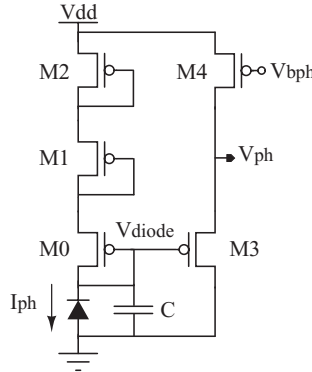


Figure 5.2: Schematic of photoreceptor.

Fig. 5.2 shows the schematic of the photoreceptor. A photodiode is used as the light sensitive element, implemented as n-well on p-sub [150,151]. Stacked diode connected pFETs convert the photocurrent I_{ph} to a voltage signal V_{diode} . All pFETs

work in the subthreshold region due to the small magnitude of the photocurrent.

The I-V relation of a pFET in subthreshold is given by

$$I_d = I_0 e^{\frac{-\kappa V_g - (1-\kappa)V_w}{V_T}} \left(e^{\frac{V_s}{V_T}} - e^{\frac{V_d}{V_T}} \right) \quad (5.2)$$

where V_s , V_g , V_d , and V_w are the source, gate, drain, and well voltage of the transistor, and I_0 is the scaling current. When $V_{sd} \geq 4V_T$, the transistor is in saturation so that V_{sd} has little influence on the current. For a diode connected pFET, the transistor is normally in saturation. With the well connected to V_{dd} , its I-V relation is simplified as

$$I_d = I_0 e^{\frac{-(1-\kappa)V_{dd}}{V_T}} e^{\frac{V_s - \kappa V_g}{V_T}} \quad (5.3)$$

Writing down the I-V relation of the three diode connected pFETs, we derive

$$V_{diode} = V_{dd} + V_T \frac{\kappa^2 + \kappa + 1}{\kappa^3} \ln \left(\frac{I_0}{I_{ph}} \right) \quad (5.4)$$

Therefore

$$dV_{diode} = -g_A V_T \frac{dI_{ph}}{I_{ph}} \quad (5.5)$$

where $g_A = \frac{\kappa^2 + \kappa + 1}{\kappa^3}$ is the gain. The reason to stack more than one transistor is to increase the gain. For one pFET, $g_A = \frac{1}{\kappa}$, and for two stacked pFETs, $g_A = \frac{\kappa + 1}{\kappa^2}$. However, increasing the number of stacked transistors reduces the voltage dynamic range and increases the size of the photoreceptor. So there is a trade-off here. As shown in (5.5), the change of V_{diode} is proportional to the contrast $\frac{dI_{ph}}{I_{ph}}$, rather than the change of the absolute illuminance. Therefore it reduces the influence from the background light intensity on the motion detection. Transistor $M3$ functions as a source follower so that V_{ph} is a voltage shifted version of V_{diode} . It also isolates the

diode from the load of the next stage. Capacitor C is partially from the the parasitic capacitance of the diode ($\sim 100\text{fF}$) and partially from an explicit capacitor ($\sim 100\text{fF}$) to filter out high frequency output. The time constant $\tau = \frac{g_A V_T C}{I_{ph}}$. For $I_{ph} = 100$ pA, $\tau \approx 3.2 \times 10^{-4}$ s, $f_{3dB} \approx 500$ Hz.

5.1.1.2 GmC Type Filter

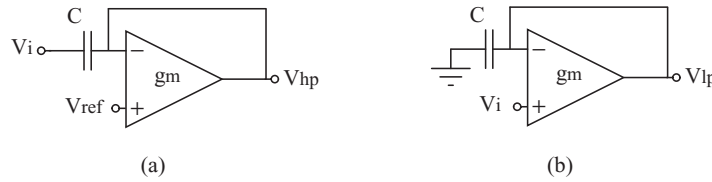


Figure 5.3: Block diagram of GmC filters: (a) high-pass filter, (b) low-pass filter.

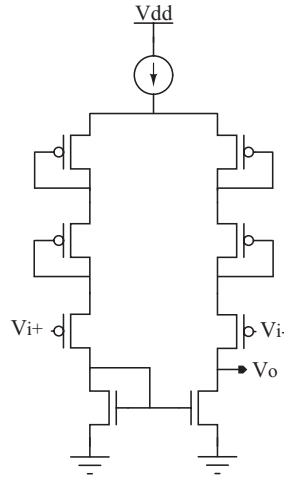


Figure 5.4: Schematic of transconductance amplifier.

Both the high-pass filter and the low-pass filter are GmC type filters [152], as shown in Fig. 5.3. The time constant is given by $\tau = \frac{C}{g_m}$. The purpose of the high-pass filter is to filter out the DC component in the signal. So the cut-off frequency needs to be as small as possible. The low-pass filter is to provide the delay between

the two branches in the EMD. Experimental data from flies shows that the time constant is around 30 ms. The transconductance amplifier used is shown in Fig. 5.4, which operates at the subthreshold region for small g_m , thus large τ . Here we use stacked diode connected pFETs to reduce g_m further. Without diode connected pFET, $g_m = \frac{I_b}{2V_T}\kappa$, with one diode connected pFET, $g_m = \frac{I_b}{2V_T}\frac{\kappa^2}{1+\kappa}$, and with two diode connected pFETs, $g_m = \frac{I_b}{2V_T}\frac{\kappa^3}{1+\kappa+\kappa^2}$. However more diode connected pFETs mean less dynamic range of the output voltage and larger area. So we pick two here. Even with the reduced g_m , to implement a large time constant filter, a large capacitor is still needed. For $\tau = 30$ ms with $I_b = 50$ pA, a capacitor of ~ 160 pF is needed.

To build high density EMD array, compact implementation of large time constant filters is necessary. There are several techniques in the literature [153–156]. We propose large time constant filters with adaptation and indirect feedback. It is described in Appendix A. For the initial proof of concept of the motion sensor, we did not implement them in the sensor. This could be pursued in the future.

5.1.1.3 Gilbert Multiplier [1]

Fig. 5.5 shows the Gilbert multiplier, which is also based on the operation at subthreshold region where

$$I_{o+} - I_{o-} = I_b \tanh \frac{\kappa(V_{i1} - V_{r1})}{2} \tanh \frac{\kappa(V_{i2} - V_{r2})}{2} \quad (5.6)$$

$$\approx \frac{1}{4} I_b \kappa^2 (V_{i1} - V_{r1})(V_{i2} - V_{r2}) \quad (5.7)$$

The approximation is good when V_{i1} is close to V_{r1} and V_{i2} is close to V_{r2} . Normally the DC voltage of V_{i2} needs to be lower than V_{i1} so that both $M1$ and $M2$ can operate at saturation region. A well control voltage V_w is used here so that V_{i1} and V_{i2} can operate at the same DC voltage level [69].

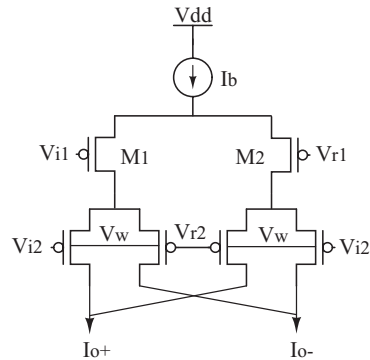


Figure 5.5: Schematic of Gilbert multiplier.

5.1.2 Mismatch

To evaluate the mismatch among the EMDs, a chip with 10 EMDs was fabricated and tested. Fig. 5.6 shows the steady state tuning response (response vs. stimulus temporal frequency) of the 10 EMDs on the same chip. The effect of mismatch on the EMD response is obvious. We will come back to the solution of mismatch in the following sections.

5.2 On-chip Spatial Filter

Once the spatial pattern of the optic flow is generated, it feeds to spatial filters for extraction of motion parameters. In general, there are two approaches to

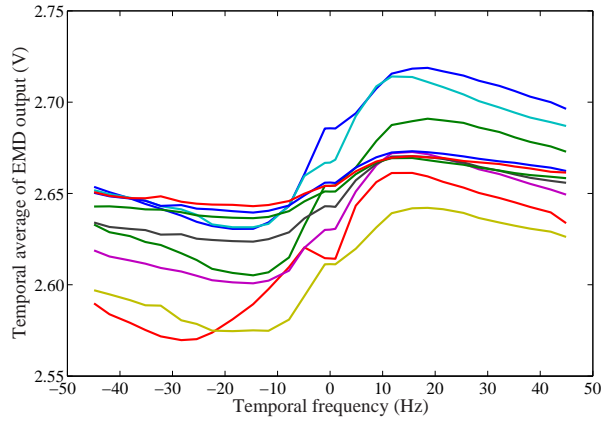


Figure 5.6: Steady state tuning curve of 10 EMDs, positive frequency: stimuli moving from right to left, negative frequency: stimuli moving from left to right.

spatial filters. One is to use networks of resistive components to generate a spatial distribution of voltages or currents, and the other is to use nonvolatile storage to store a spatial pattern of voltages or currents. In both cases, the filter operation is essentially a vector dot product for a single filter and a vector matrix multiplication for multiple filters. Similar operation has been used in other applications under different names, such as matched filter, correlation, and similarity evaluation. The filter output indicates how closely the input pattern matches the filter pattern. The filter can be implemented in either digital domain or analog domain. Hahm [157] compared digital and analog approaches using power consumption as the primary metric. It is found that analog circuit implementation is more power efficient for shorter and faster filters, while digital circuit is more power efficient for longer and slower filters. Digital implementation also has the advantage of higher level of precision. In our work, we focus on analog implementation mainly for its low complexity, low power consumption, and continuous operation in time.

5.2.1 Resistive Networks based Implementation

Resistive networks have been widely used for spatial filters [1, 158]. In VLSI technology, the resistive components in the networks could be implemented by real resistors, transistors operating in linear region, or more complex active components functioning as resistors. The detailed treatment of resistive networks based on pure resistors can be found in [1]. Mahowald [159] used two dimension resistor networks to emulate the lateral information spread operation performed by horizontal cells in retina. Kobayashi [160] used active resistor networks to implement a two dimensional Gaussian shape spatial filter for image smoothing. Choi and Abidi [161] used resistor averaging networks in the A/D converter to lower the impact of the offsets, therefore to improve the accuracy. It could also help improving the speed. Andreou and Boahen [162] proposed a current-mode diffusor network using MOS transistors in subthreshold. It has the same network response as the resistive network with the additional benefit of adjustable diffusion length by gate voltages. Often resistive networks based spatial filter gives predefined shape and there is little flexibility for change once it is hardwired in the chip. Filters with special shape are hard to implement. It is not always the case that the configuration for a specific shape can be found.

In the motion image sensor for WFI based navigation, sinusoidal spatial filters are needed. There is no reported solution for such a filter based on resistive networks. We first synthesize a sinusoidal function element that takes angles as input represented in electrical signals and outputs the sinusoidal value of the angles in

voltages or currents; then we assembly a sinusoidal spatial filter with the resistive networks and the sinusoidal function elements.

5.2.1.1 Sinusoidal Function Circuit

Gilbert [163] has proposed a method for analog synthesis of trigonometric functions using bipolar transistors. Since the I-V characteristics of MOSFETs in subthreshold is very similar to bipolar transistors, we can easily adapt the method to implement trigonometric functions in MOSFET. Fig. 5.7 shows simplest network to implement a $\sin(\cdot)$ function. The angular input $x = \frac{\pi(V_+ - V_-)}{3E_B}$ in radians, where

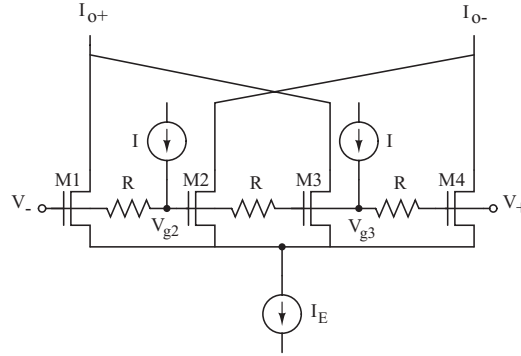


Figure 5.7: Schematic of a sinusoidal function circuit.

$E_B = IR$. The differential output $I_o = I_{o+} - I_{o-} = \eta I_E \sin(x)$, where the efficiency $\eta = 2e^{-\frac{\pi^2}{2\kappa\alpha}}$ with $\alpha = \frac{IR}{V_T}$. The two current sources I , resistor chain, and the input voltages create the gate voltage differences of transistor $M1-M4$. These transistors have common source voltages so that the ratio of their channel currents is $e^{\frac{\kappa V_-}{V_T}} : e^{\frac{\kappa V_{g2}}{V_T}} : e^{\frac{\kappa V_{g3}}{V_T}} : e^{\frac{\kappa V_+}{V_T}}$. Therefore the output current is

$$I_o = I_{o+} - I_{o-} = I_E \frac{e^{\frac{\kappa V_-}{V_T}} + e^{\frac{\kappa V_{g3}}{V_T}} - e^{\frac{\kappa V_{g2}}{V_T}} - e^{\frac{\kappa V_+}{V_T}}}{e^{\frac{\kappa V_-}{V_T}} + e^{\frac{\kappa V_{g2}}{V_T}} + e^{\frac{\kappa V_{g3}}{V_T}} + e^{\frac{\kappa V_+}{V_T}}} \quad (5.8)$$

This is the simplest case of a general network with N identical transistors, $N - 1$ resistors R , and $N - 2$ equal current sources I configured in the similar way. It has been shown the network performs the exact $\sin(\cdot)$ function as $N \rightarrow \infty$ [163]. For N being as small as 4, the network is still reasonably close to a $\sin(\cdot)$ function, especially for angle $|x| < \frac{\pi}{2}$. Fig. 5.8 shows the differential current output I_o for the DC sweep of V_{offset} from -0.15 V to 0.15 V, where $V_{cm} = 2$ V, $V_+ = V_{cm} + V_{offset}$, and $V_- = V_{cm} - V_{offset}$. With $R = 10$ k Ω , $I = 10$ μ A, $I_E = 20$ nA, the offset spans the equivalent angular input $-\pi$ to π .

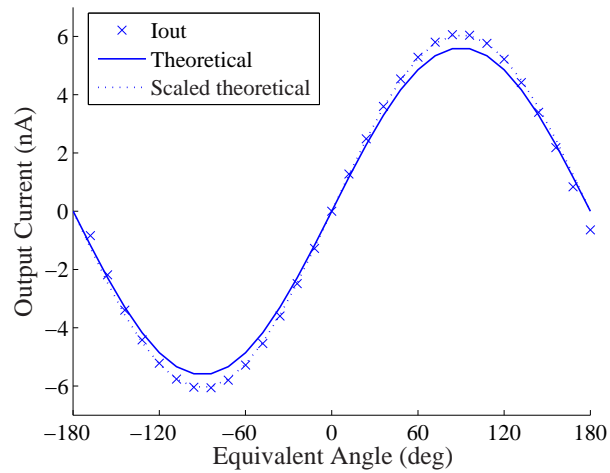


Figure 5.8: Differential current output I_o of the $\sin(\cdot)$ function circuit for a DC sweep simulation of the input offset V_{offset} . The theoretical approximation is $5.61 \sin(x)$ with $\kappa = 0.65$. The scaled theoretical approximation is $6.06 \sin(x)$, obtained by scaling $\sin(x)$ with the magnitude of the sine function from the simulation.

5.2.1.2 Sinusoidal Spatial Filter

Fig. 5.9 shows how to implement the spatial sinusoidal filter with resistor networks and the $\sin(\cdot)$ circuit. The resistor network generates an evenly spaced voltage distribution. Each voltage node in the resistor network and the common voltage node V_c form the differential voltage input pair to a $\sin(\cdot)$ function circuit. I_{Ei} are the input currents to be filtered. The scaling of the input current of I_{Ei} by the $\sin(\cdot)$ circuit performs the multiplication naturally. I_{oi} are the output currents at each location. For n -point filter, there are $n - 1$ resistors between V_l and V_h . The angular range of the filter is adjustable, given by $\frac{\pi(V_h - V_l)}{3E_B}$ in radians. The sampling interval is $\frac{\pi(V_h - V_l)}{3(n-1)E_B}$ in radians.

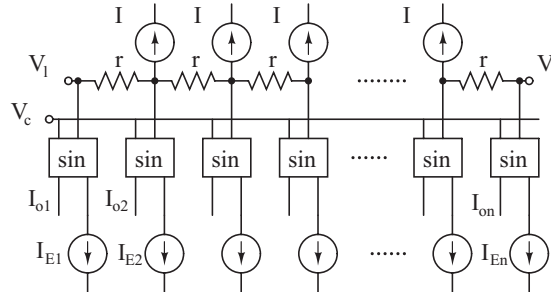


Figure 5.9: Diagram of sinusoidal network circuit.

The current sources I connected to the resistor networks compensate the current sources within the $\sin(\cdot)$ circuits. Fig. 5.10(a) shows the entire resistor networks with the resistors and current sources inside and outside the $\sin(\cdot)$ circuits. In the single branch shown in Fig. 5.10(b), it can be derived that voltage $V_o = \frac{V_l \cdot 3Rr + V_h \cdot 3Rr + V_c \cdot r^2}{6Rr + r^2}$, which is not affected by I . When $R \gg r$, $V_o \approx \frac{V_l + V_h}{2}$. Similarly it can be shown that in the full networks, the voltages along the resistor

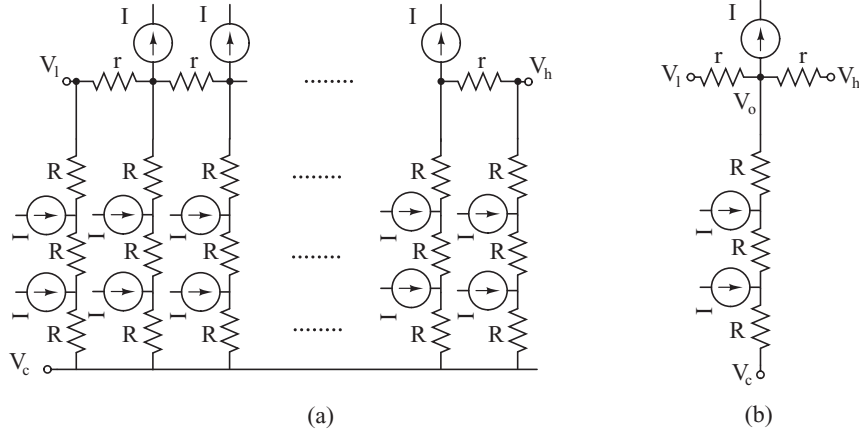


Figure 5.10: Resistor networks and current sources in the sinusoidal spatial filter circuit, (a) the full networks, (b) a single branch.

network between V_i and V_h are not affected by the current sources. And they are evenly distributed from V_i to V_h when $R \gg r$.

Fig. 5.11 shows the simulation result of a 10-element sinusoidal filter with $r = 100 \Omega$, $R = 10 \text{ k}\Omega$, $I = 10 \mu\text{A}$, and $I_{Ei} = 20 \text{ nA}$. For AMI $0.5 \mu\text{m}$ technology, the resistor can be realized by poly2 layer with high resistivity, $\sim 949 \Omega/\text{sq}$. According to SCMOS layout rules, both $10 \text{ k}\Omega$ resistor and 100Ω resistor require at least $\sim 50\lambda \times 7\lambda \approx 43\mu\text{m}^2$ chip area each.

5.2.2 Nonvolatile Storage based Approach

For nonvolatile storage based approaches, each coefficient is individually programmed and stored as nonvolatile charges. So there is great flexibility to implement filters with variety of shapes. The filter can also be modified during operation after programming, making it adaptive. Floating gate MOSFET has been widely used as the nonvolatile storage unit in CMOS technology. The charges at the floating

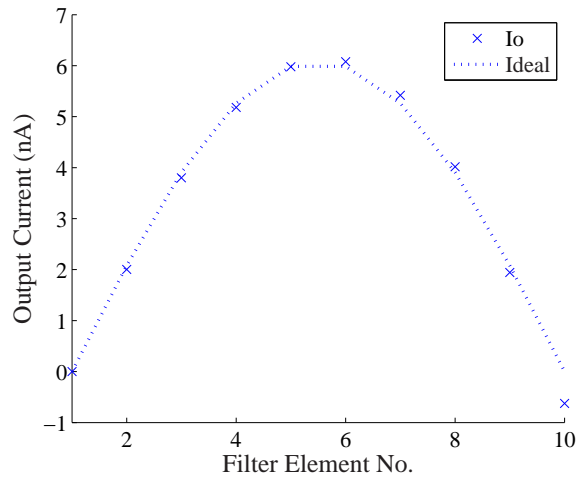


Figure 5.11: Simulation of the sinusoidal network circuit.

gate can be programmed by charge injection or tunnelling. Floating gate MOS-FET can be easily implemented in double-poly CMOS process, while it has also been demonstrated in single polysilicon CMOS process [164]. Details about injection and tunnelling can be found in [27]. Charge injection and tunnelling effect have been used in many applications [28, 32]. A programmable analog memory-cell array was described in [36]. Bandyopadhyay et al. [38] used a programmable floating-gate array (PFGA) to implement matrix transformations on images. The I-V relation of floating-gate transistors was first measured and stored as a lookup table. To program a pre-defined floating gate voltage, the channel current of the transistor was monitored and compared with the data in the lookup table. Optimization was performed to find the best pulse number and pulse width to inject electrons to the floating gate. This way, high programming accuracy was achieved (maximum error $< 0.2\%$). PFGA technology is a flexible way to implement filters of various coefficients and shapes. However, programming using the PFGA requires

substantial system infrastructure and support. The approach we proposed takes advantage of the intrinsic negative feedback to program the current. It is direct and simple. Mismatch compensation for the stages before filtering operation can be easily incorporated.

Ultra-violet (UV) illumination can make silicon-dioxide slightly conductive, therefore can also be used to add or remove charges on the floating gate depending on the potential setup. An adaptive four quadrant matrix-vector multiplier has been reported using UV illumination for analog hardware neural network [165]. When multiple voltage inputs are coupled to the floating gate through capacitors, match filter can also be implemented without programming the floating gate with charges, rather by coupling combinations of input voltages. Yamasaki [166] reported an analog similarity evaluation circuit with variable functional forms using this technique.

In the proposed motion sensor, we use floating-gate MOSFET to implement the spatial filter. Compared with the resistive networks approach, it is more compact and also provides us the means to compensate the mismatch from previous stages.

5.2.2.1 P-type Floating Gate MOSFET

P-type floating gate MOSFET is often used due to its easy setup for injection current. Fig. 5.12 shows the circuit model for a p-type floating gate MOSFET. It is also called floating gate pFET. The resistor R and the voltage controlled voltage source $E = V(V_{fg}, 0)$ is put here just to introduce a DC path from the floating node

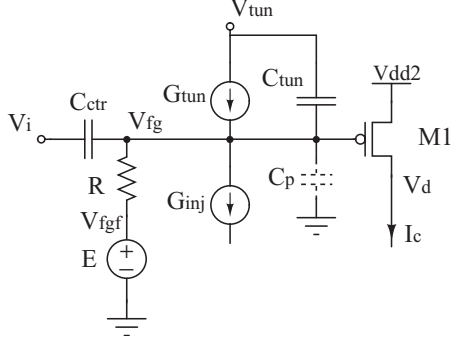


Figure 5.12: Circuit model of a p-type floating gate MOSFET.

V_{fg} to ground, therefore to make the simulation convergence easier. There is no current flowing through R . The input V_i from the control gate is coupled to the floating gate V_{fg} through C_{ctr} . Charge can be added to the floating node by Fowler-Nordheim tunnelling current G_{tun} from tunnelling node V_{tun} . It is implemented by a pFET where its gate is connected to the floating node and its source, drain, and body are connected to V_{tun} . According to an empirical model the tunnelling current is given by [167]

$$G_{tun} = I_{tun0}WL \exp\left(-\frac{V_c}{V_{ox}}\right) \quad (5.9)$$

where I_{tun0} is the scaling tunnelling current, V_c is a constant depending on the oxide thickness, V_{ox} is the voltage across the silicon-dioxide. When V_{tun} is high enough, the tunnelling current G_{tun} adds charges to the floating gate. A MOS capacitor C_{tun} is introduced by the tunnelling structure. Thus, besides adding charges to the floating gate, V_{tun} can also function as a secondary control gate to modulate the floating gate voltage through capacitor coupling. C_p models the parasitic capacitance between the gate and the source C_{gs} , the gate and the drain C_{gd} , and the gate and the body

C_{gb} . Normally $C_{ctr} \gg C_{tun}$ and $C_{ctr} \gg C_p$, so that

$$dV_{fg} = dV_i \frac{C_{ctr}}{C_{tol}} = dV_i \frac{C_{ctr}}{C_{ctr} + C_{tun} + C_p} \approx dV_i \quad (5.10)$$

The charges on the floating gate are removed by injection current G_{inj} . For floating gate pFET, the injection current is caused by impact ionized hot electron injection. Channel holes are accelerated by channel to drain electric field and obtain energy high enough to promote electrons to conduction band when colliding with electron-hole pairs; freed electrons are pushed backwards from drain to channel by the same field and are injected into oxide if they have higher kinetic energy than 3.1 eV; these electrons can be collected by the floating gate if the potential of the floating gate is higher than the channel near the drain. A semi-empirical equation for the injection current is given by [167]

$$G_{inj} = \alpha I_c \exp \left(-\frac{\beta}{(V_{gd} + \delta)^2} + \lambda V_{sd} \right) \quad (5.11)$$

where I_c is the channel current of the transistor, V_{sd} is its source to drain voltage, and V_{gd} is the floating gate to drain voltage. α , β , and δ are fit parameters. The injection current is linearly proportional to the channel current, exponentially proportional to V_{sd} , and has a more convoluted relation with V_{gd} . For injection to happen, all three conditions, enough channel current, enough V_{gd} , and enough V_{sd} need to be satisfied.

We use V_{tun} to set the initial floating gate voltage so that its I_c is smaller than the target current I_t to program. G_{inj} is configured in a negative feedback to program I_t as the channel current of the transistor. I_t is provided as a bias current at V_d . Initially I_c is less than I_t , so V_d is pulled down, which will provide enough

V_{gd} and V_{sd} for injection to happen. The injected electrons onto the floating gate lower the floating gate potential, therefore increase I_c . This procedure continues until I_c is close to I_t enough so that V_d increases and eventually shuts off G_{inj} . This is also referred as the constant current configuration [31], and has been used in autozeroing amplifier [29] and fixed pattern noise reduction in imager [32]. To facilitate the injection, V_{dd2} is raised during programming phase and lowered during normal operation.

5.2.2.2 Basic Mechanism for Mismatch Compensation and Filter Programming

By programming a target current to the floating gate pFET, we can achieve mismatch compensation and filter programming, which is enabled by the exponential I-V relationship of pFET in subthreshold. For the pFET $M1$ in subthreshold and saturation in Fig. 5.12,

$$I_c = I_0 \exp\left(\frac{\kappa(V_{dd2} - V_{fg})}{V_T}\right) \quad (5.12)$$

$$V_{fg} = V_i \frac{C_{ctr}}{C_{tot}} + \frac{\Delta Q_{tot}}{C_{tot}} \quad (5.13)$$

V_i is the output from the sensor front-end. Suppose all sensor units receive the same input from the outside world, we should get the same current output from the pFETs assuming there is no charge on the floating gate $\Delta Q_{tot} = 0$. However because of the fabrication mismatch, the sensor units produce different outputs, and the floating gates have different initial charges. Altogether the current outputs from the floating gate pFET are all different. The initial random charges on the

floating gate do contribute more mismatch compared with the regular pFET. But as it will be shown, the ability to change the charges on the floating gate enables us to compensate the mismatch and to programm filters.

For a given common input to all sensor unit, if we know that I_{common} should be the current output, we can use I_{common} as the constant current bias for the floating gate pFETs, therefore to program their channel currents to I_{common} . In the situation where we don't know the supposed current output, we can just pick an I_{common} in the subthreshold. The difference will only introduce a constant scaler for all units.

I_{common} is written as

$$I_{common} = I_0 \exp \left(\frac{\kappa(V_{dd2} - V_i \frac{C_{ctr}}{C_{tot}} - \frac{\Delta Q_{init}}{C_{tot}} - \frac{\Delta Q_{adapt}}{C_{tot}})}{V_T} \right) \quad (5.14)$$

where ΔQ_{init} is the initial charge on the floating gate and ΔQ_{adapt} is the charge added to compensate the mismatch. Now for the same input, the same output is obtained, therefore the mismatch is cancelled by the charge ΔQ_{adapt} .

We can now program the filter coefficient F_c . We use $I_{filter} = F_c \cdot I_{common}$ as the bias current to program, thus

$$I_{filter} = I_0 \exp \left(\frac{\kappa(V_{dd2} - V_i \frac{C_{ctr}}{C_{tot}} - \frac{\Delta Q_{init}}{C_{tot}} - \frac{\Delta Q_{adapt}}{C_{tot}} - \frac{\Delta Q_{filter}}{C_{tot}})}{V_T} \right) \quad (5.15)$$

$$= I_{common} \exp \left(\frac{\kappa \left(-\frac{\Delta Q_{filter}}{C_{tot}} \right)}{V_T} \right) \quad (5.16)$$

$$F_c = \exp \left(\frac{\kappa \left(-\frac{\Delta Q_{filter}}{C_{tot}} \right)}{V_T} \right) \quad (5.17)$$

where ΔQ_{filter} is the charges added to program the coefficient F_c . Notice that we can only program a current that is larger than the existing channel current, which means that we can only program $F_c \geq 1$. However, when used in differential mode,

the method does not limit the range of the filter coefficient that can be programmed as shown in (5.23) and (5.24).

During normal operation with the input V_i , the current output I_o is

$$I_o = I_0 \exp\left(\frac{\kappa(V_{dd2} - V_i \frac{C_{ctr}}{C_{tot}} - \frac{\Delta Q_{init}}{C_{tot}} - \frac{\Delta Q_{adapt}}{C_{tot}} - \frac{\Delta Q_{filter}}{C_{tot}})}{V_T}\right) \quad (5.18)$$

$$= I_0 \exp\left(\frac{\kappa(V_{dd2} - V_i \frac{C_{ctr}}{C_{tot}} - \frac{\Delta Q_{init}}{C_{tot}} - \frac{\Delta Q_{adapt}}{C_{tot}})}{V_T}\right) \exp\left(\frac{\kappa(-\frac{\Delta Q_{filter}}{C_{tot}})}{V_T}\right) \quad (5.19)$$

$$= I_i \frac{I_{filter}}{I_{common}} \quad (5.20)$$

$$= I_i \cdot F_c \quad (5.21)$$

where I_i is the compensated output current, and I_o is the compensated and filtered output. So filtering operation is achieved by this floating gate pFET as well. Here the mismatch compensation and filter programming are shown to be a two step procedure. They can be achieved in one step to program $I_{filter} = F_c I_{common}$ directly,

$$I_{filter} = I_0 \exp\left(\frac{\kappa(V_{dd2} - V_i \frac{C_{ctr}}{C_{tot}} - \frac{\Delta Q_{init}}{C_{tot}} - \frac{\Delta Q_{prog}}{C_{tot}})}{V_T}\right) \quad (5.22)$$

where $\Delta Q_{prog} = \Delta Q_{adapt} + \Delta Q_{filter}$.

In order to implement the four quadrant multiplication for the filtering operation, both the input I_i and the coefficient F_c need to be differential signals. The four quadrant multiplication breaks into four one quadrant multiplication such that four floating gate pFETs implement one coefficient.

$$I_i F_c = (I_+ - I_-)(F_p - F_m) = (I_+ F_p + I_- F_m) - (I_- F_p + I_+ F_m) \quad (5.23)$$

where $I_+, I_- > 0$ and $F_p, F_m \geq 1$,

$$\begin{cases} F_c > 0 : & F_p = 1 + F_c, & F_m = 1 \\ F_c < 0 : & F_p = 1, & F_m = 1 - F_c \\ F_c = 0 : & F_p = 1, & F_m = 1 \end{cases} \quad (5.24)$$

Because we use the subthreshold I-V characteristics of the transistors, we need to maintain the magnitude of the channel current in the range of subthreshold. Thus the range of F_c is limited by the channel current range in subthreshold operation.

5.2.2.3 Programmable Current Element (PCE)

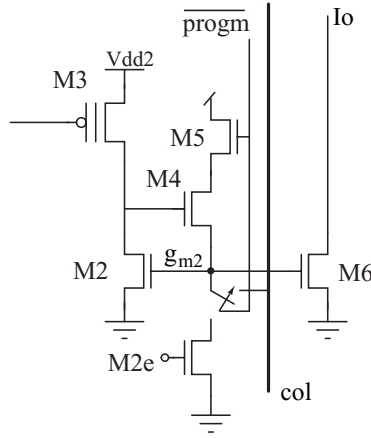


Figure 5.13: Illustration of the programmable current element (PCE).

Fig. 5.13 illustrates the basic circuits for the programmable current element (PCE), where mismatch compensation, filter programming, and filtering operation can all be achieved at one place. Similar structure has been used for fixed pattern noise compensation in CMOS imagers [30, 32]. $M3$ is the floating gate pFET to store charges for filter coefficient and mismatch compensation, and to perform multiplication. When \overline{prog} is low, transistor $M5$ is turned off and node g_{m2} is

connected to the column line col . The voltage on the col is generated by a current bias and $M2$ mirrors the bias current to $M3$. This bias current is programmed onto $M3$ by the mechanism explained above. Different currents can be programmed into different PCEs as the filter coefficients or same currents can be programmed across all PCEs under the same input to compensate mismatch.

When $\overline{prog_m}$ is high, $M5$ is turned on and g_{m2} is connected to the bias transistor $M2e$. Together, $M5$, $M4$, $M2$, and $M2e$ function as a current conveyor. The current from $M3$ is mirrored to $M6$ and I_o is the output current. $M3$ performs the multiplication operation and the charge stored on its floating gate functions as the coefficient of the filter. So each PCE performs multiplication between one input element and one filter coefficient, an array of PCEs with their output tied together performs the dot product between the filter and the input vector, i.e. the spatial filtering of the input vector.

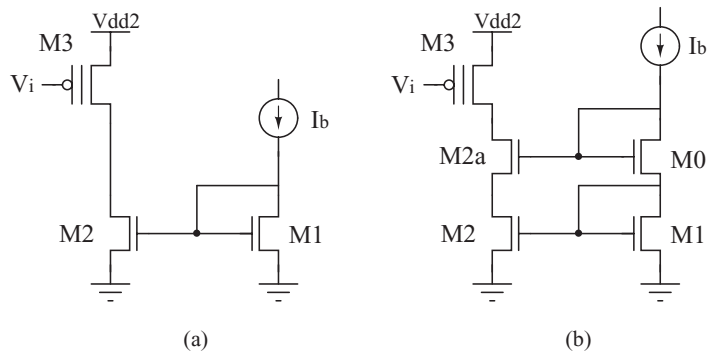


Figure 5.14: Current bias for the floating-gate pFET, (a) simple current mirror, (b) cascode current mirror

To program a target current to the floating gate pFET, it is important to mirror the current accurately to the drain of the floating gate pFET. We compared

current bias using the simple current mirror and the cascode mirror in Fig. 5.14. We use HSPICE to simulate the programming process with the floating gate pFET model in Fig. 5.12. Transient sweep simulation is performed for the programming of currents from 10 nA to 100 nA. The programmed currents at the equilibrium state are plotted in Fig. 5.15. It shows that injection occurs with the cascode and the programming accuracy is much improved compared with the simple current mirror.

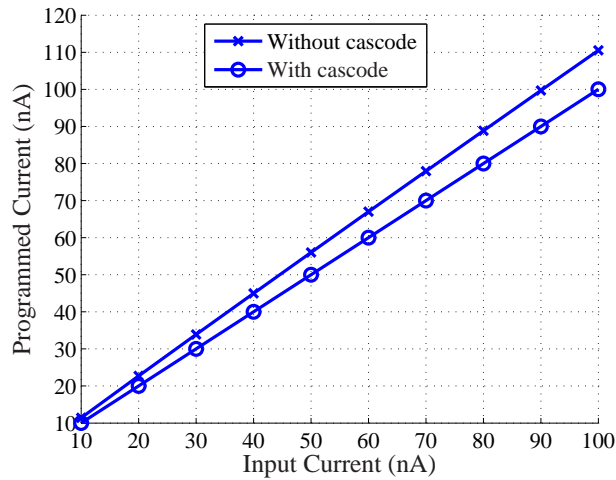


Figure 5.15: Programmed current at equilibrium state from transient sweep simulation.

Fig. 5.16 shows the actual schematic of the PCE with 11 transistors. Cascode with fixed bias is added to improve both the programming accuracy and the output mirror accuracy, at the same time to maintain the easy control of the circuit. A diode connected nFET $M4a$ is added to the branch of $M4$ and $M5$ to shift the drain voltage of $M3$. This brings the drain voltage of $M3$ closer before and after switching from the programming mode to the filtering mode, thus reduces early effect and parasitic coupling. This way, the output current is closer to the current

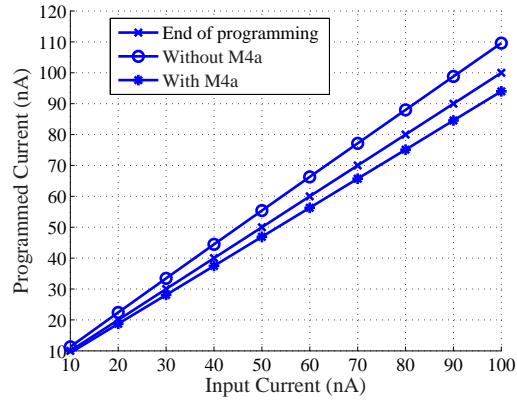


Figure 5.17: Transient sweep simulation of programming PCE, \times : the channel current in $M3$ after programming is done, \circ : the channel current in $M3$ after switching to filtering mode for PCE without $M4a$, $*$: the channel current in $M3$ after switching to filtering mode for PCE with $M4a$. The output current in $M7$ is very close to the channel current in $M3$ in the filtering mode.

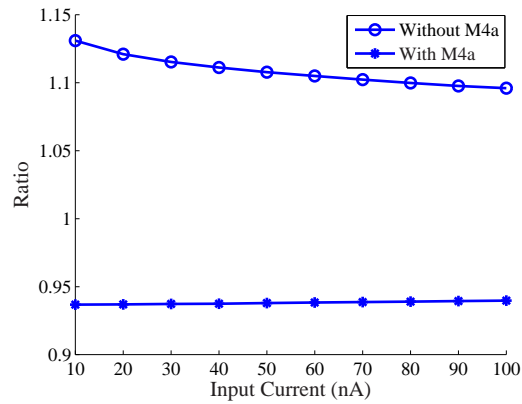


Figure 5.18: Ratio of the output current to the target current after programming in the filtering mode from transient sweep simulation of PCE.

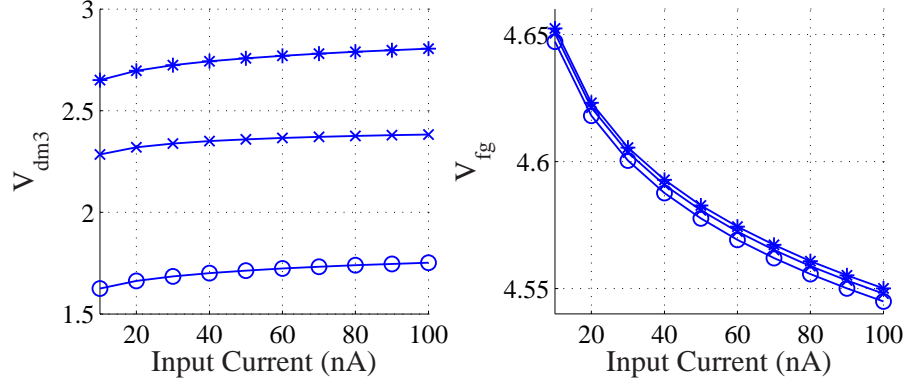


Figure 5.19: Drain voltage (left) and floating gate voltage (right) from transient sweep simulation of PCE. \times : when programming is finished, \circ : after switching to filtering mode for PCE without $M4a$, $*$: after switching to filtering mode for PCE with $M4a$.

program the PCE. Similar structure is used to match the PCE. $gm1$ is connected to $gm2$ in the PCE through col when $prog_m$ is high. Fig. 5.21 shows the layout of the PCE with the pitch size of $70 \mu m \times 74.2 \mu m$.

5.3 Motion Image Sensor Architecture and Operation

Fig. 5.22 shows the architecture of the motion image sensor. The front end of the sensor is a one-dimension photoreceptor array that converts light into voltage signal. The one-dimension EMD array computes the optic flow along the horizontal direction. As the proof of concept, we intend to use the sensor for ground vehicle navigation. Similar approach can be extended to two dimensions. The differential output currents from the EMD array feed into the programmable current matrix (PCM) where mismatch compensation, filter programming, and filtering are

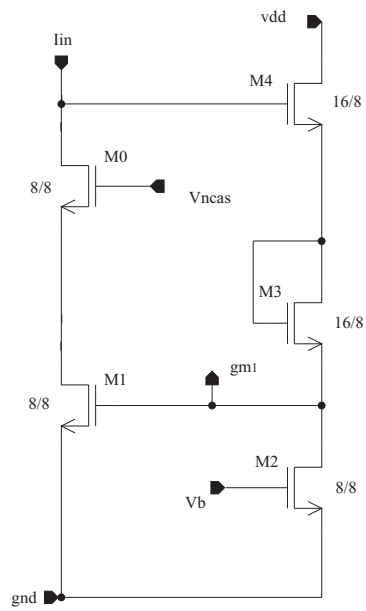


Figure 5.20: Schematic of the bias circuit for the PCE.

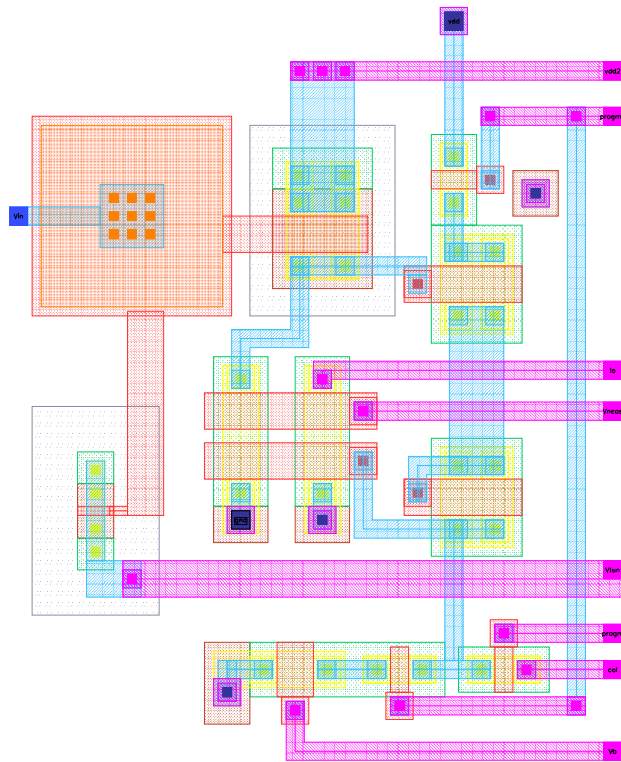


Figure 5.21: The layout of the PCE.

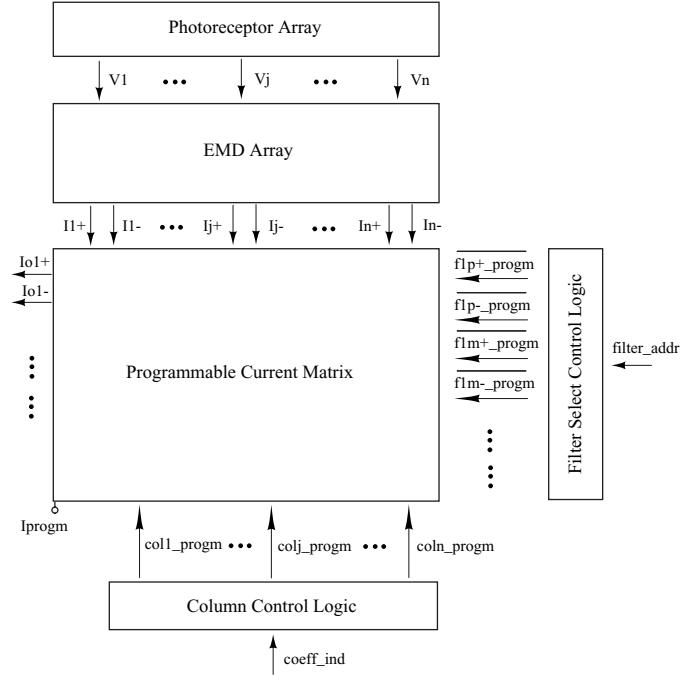


Figure 5.22: Architecture of the motion image sensor.

performed. The filter select control logic chooses which filter to program and the column control logic chooses which coefficient to program.

5.3.1 Programmable Current Matrix (PCM)

PCM is the post-sensor computation core, where mismatch compensation, filter programming, and filtering are performed. It consists of a matrix of the PCE. Fig. 5.23 illustrates the PCM. Differential current outputs from EMD array are converted to differential voltages by diode connected pFETs and feed to PCM, specifically, the control gates of the floating gate pFETs in the PCEs. Each coefficient block in the PCM is a PCE. The dot product between two differential vectors

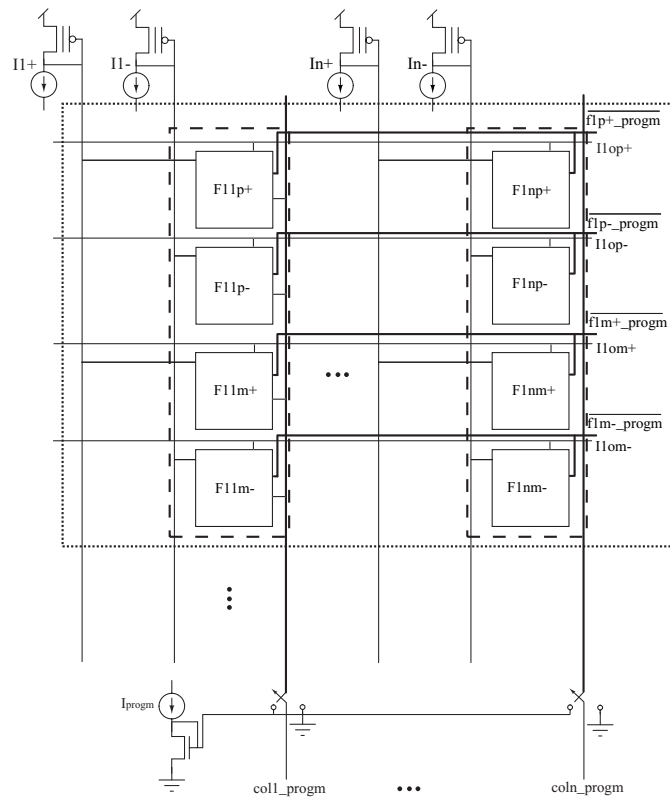


Figure 5.23: Illustration of the PCM.

can be broken into four dot products of vectors in the first quadrant.

$$\vec{I}_{diff} \cdot \vec{F}_{diff} = (\vec{I}_{i+} \cdot \vec{F}_{p+} + \vec{I}_{i-} \cdot \vec{F}_{m-}) - (\vec{I}_{i-} \cdot \vec{F}_{p-} + \vec{I}_{i+} \cdot \vec{F}_{m+}) \quad (5.25)$$

$$= (I_{op+} + I_{om-}) - (I_{op-} + I_{om+}) \quad (5.26)$$

The subscript "+/-" in \vec{F} indicates which part of the input vector (positive or negative) the filter takes, and the subscript "p/m" indicates which part of the filter (positive or negative) it represents. Normally, $\vec{F}_{p+} = \vec{F}_{p-}$ and $\vec{F}_{m+} = \vec{F}_{m-}$, thus we should be able to program the same value to the pair at the same time. However, each PCE has its own current mirror to output the current, there is the fabrication mismatch here as well. We implement the four quadrant filter as four individual filters in the first quadrant so that we can compensate the current mirror mismatch as well. In Fig. 5.23, the implementation of one four quadrant filter is shown inside the dotted box. Four blocks in the dashed block implement the first four quadrant coefficient. Each row implements one first quadrant filter. They can all be indexed together and controlled by the filter select control logic.

5.3.2 Column Control Logic

The column control logic determines the access to the PCE. It uses a binary decoder to select one column at a time. When the column is selected, the column line is connected to the gate voltage ($gm1$ in Fig. 5.20) so that the voltage bias required to program the PCE can be passed to the PCE in the programming mode. The same select signal also selects which PCE outputs its current to the output line so that output from individual PCE can be monitored during programming. It also

helps testing and debugging. When not selected, the column line is connected to the ground. The decoder uses low active logic. Pull up transistors deactivate the line when it is not selected. Pull down transistors are also used to select all columns for mismatch compensation. In filtering mode, all columns need to be active as well so that all PCEs in a row contribute to the filter output current.

5.3.3 Filter Select Control Logic

The filter select control logic determines the operation mode of the sensor and selects the filter to program. Similar to the column control logic, it uses a binary decoder to activate one output at a time and it uses low active logic. It also has additional controls so that all outputs can be deactivated or activated. When one output is active, the selected filter is in the programming mode, its $\overline{prog_m}$ is low. When all outputs are active, the sensor is in the mismatch compensation mode so that all the PCEs are in the programming mode. When none is active, the sensor is in the filtering mode.

5.3.4 Operation Mode

The motion image sensor has three operation mode: mismatch compensation, filter programming, and filtering. We will explain the programming mode first since mismatch compensation is a special case of programming mode.

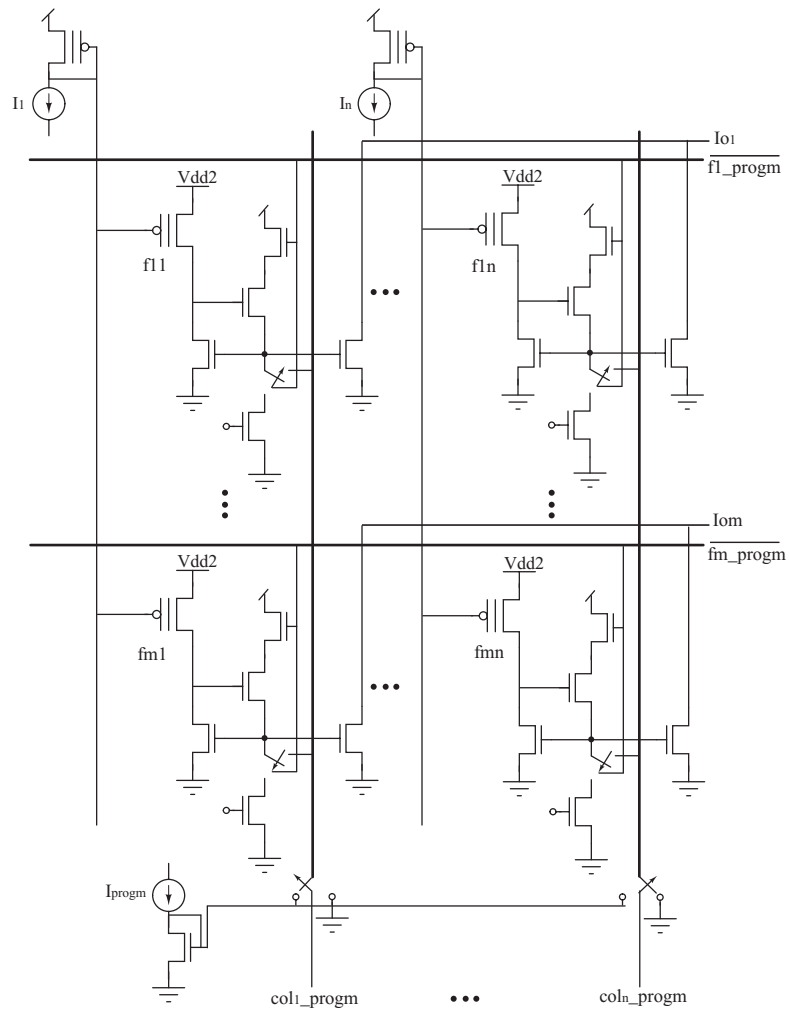


Figure 5.24: PCM in programming mode.

5.3.4.1 Filter Programming

In the filter programming mode, $\overline{f_i-progm}$ is active low one at a time so that all PCEs in the addressed filter are in the programming mode. Then $col_j-progm$ is active one at a time so that the selected column line is connected to the bias voltage. As shown in Fig. 5.24, the first filter and the first column is selected. The target current to program is mirrored as the drain bias current for the floating gate pFET in the selected PCE. And the corresponding coefficient is programmed. Filter by filter and coefficient by coefficient, all filters are programmed.

5.3.4.2 Mismatch Compensation

In the mismatch compensation mode, all $\overline{f_i-progm}$ are low and all $col_j-progm$ are high, so all PCEs are in programming mode with their gate connected to the same voltage bias converted from the target current. With the same optical input to the EMDs, the outputs from all PCEs are programmed to the same current output, therefore fabrication mismatch is cancelled.

5.3.4.3 Sensing and Filtering

As shown in Fig. 5.25, in normal filtering operation mode, all $\overline{f_i-progm}$ are high so that current conveyors of all PCEs are connected to their own current sink. Current output from each current conveyor is the compensated and filtered EMD output current. Output currents from PCEs in the same row pass to the output line I_{oi} . The total output current is the filter output current. As explained before, four

such currents are combined to obtain the output current for a four quadrant filter output.

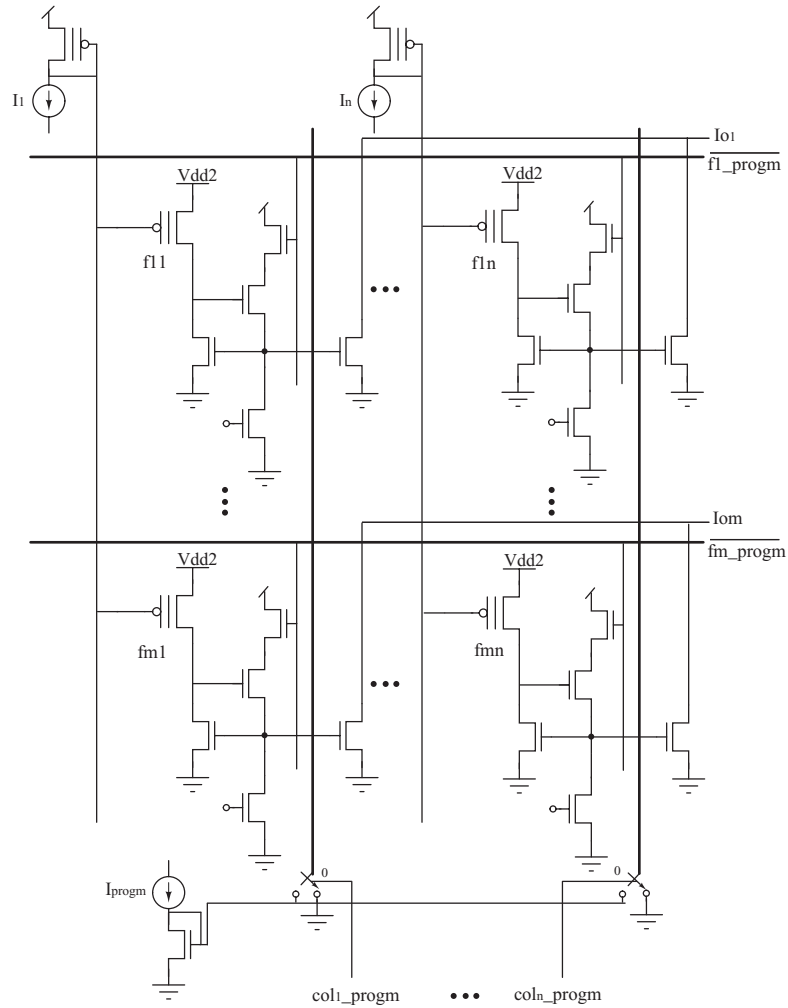


Figure 5.25: PCM in sensing and filtering mode.

5.4 Experimental Results

The circuits have been fabricated in a commercially-available $0.5 \mu\text{m}$ CMOS process with 2 polysilicon layers and 3 metal layers.

5.4.1 PCE Test

To validate the idea of the programming structure, we tested an array of 19 PCEs connected to the same voltage input at their control gates in the chip PCM. Fig. 5.26 shows the test setup for the PCM. The current to program I_{target} is provided by a source meter, which is controlled by the computer through general purpose interface bus (GPIB). The output from PCEs I_o is monitored by another source meter, and the data is collected by the computer through GPIB. The device under test (DUT), the PCM, is controlled by the data acquisition (DAQ) for the operation mode selection and the PCE access. CV meter provides the step voltage to gradually increase or decrease the V_{tun} , to start or stop the tunnelling. We first

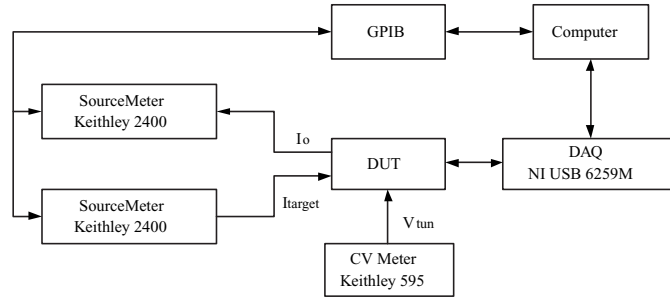


Figure 5.26: Test setup for the PCM.

tried to program a constant current to all PCEs with $I_{target} = 10$ nA. At the end of programming, the output current from each PCE I_{pgm} was quite different, see the programmed current in Fig. 5.27. We then individually measured the output current when the output current mirror was biased directly by the same input bias voltage converted from I_{target} . We call this current calibration current I_{cal} . The calibration current from each PCE was quite different as well. This comes from the

mismatch among output current mirrors. Interestingly, the programmed currents are a scaled version of the calibration currents, as shown in Fig. 5.27. Therefore we can attribute all the mismatch from the current mirror to a scaler p , and define an injection efficiency parameter k_{inj} as the percentage of the drain bias current that is programmed to the floating gate pFET. We write

$$I_{pgm} = p \cdot k_{inj} \cdot I_{target} \quad (5.27)$$

$$I_{cal} = p \cdot I_{target} \quad (5.28)$$

The ratio $\frac{I_{pgm}}{I_{cal}}$ gives us the injection efficiency, where $\mu(k_{inj}) = 0.7323$, $\delta(k_{inj}) = 0.004$, $\frac{\delta(k_{inj})}{\mu(k_{inj})} = 0.55\%$.

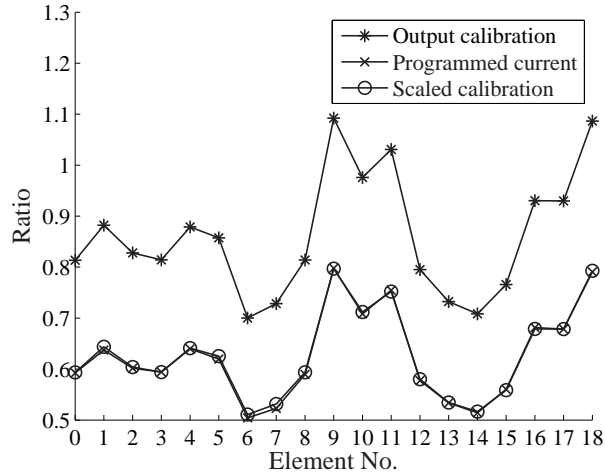


Figure 5.27: Programming of 10 nA to an array of PCEs.

Because we use the current output from the PCEs as the final output of the computation, we should compensate the mismatch at the very end of the computation, namely, the output current. We calibrate the current mirror and use it to scale the target current to program. We also use k_{inj} to scale I_{target} to obtain the desired

output current level. This way, both mismatch from sensor frontend and PCEs are compensated. Fig. 5.28 shows the calibration current to target current ratio as a function of the target current for 19 EMDs. For input current below 10 nA the ratio is very sensitive to the input current level, but above 10 nA, the variation is much smaller.

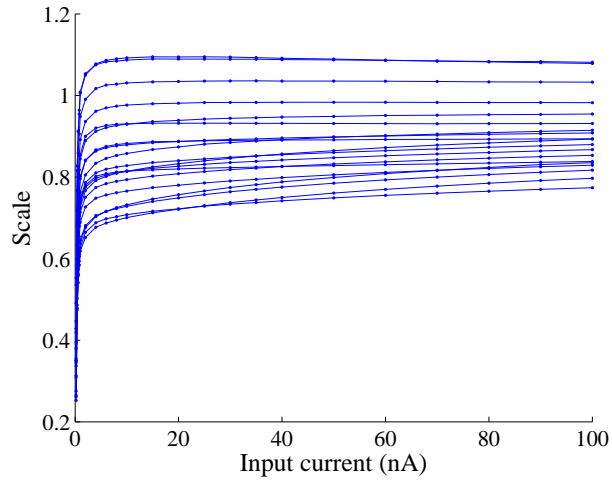
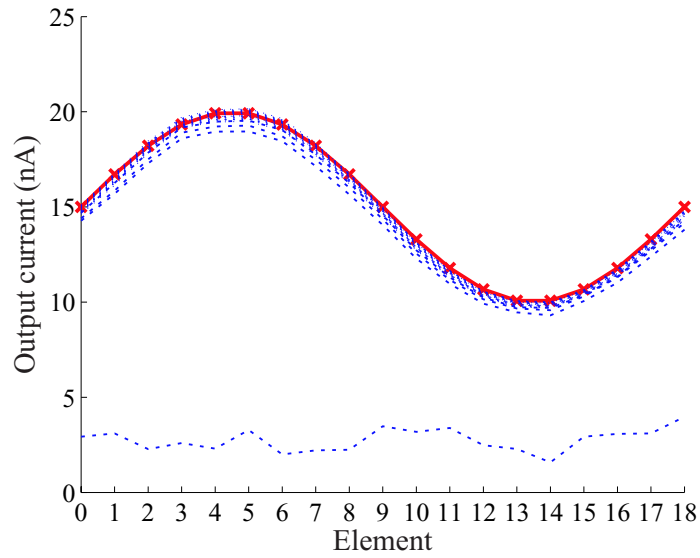


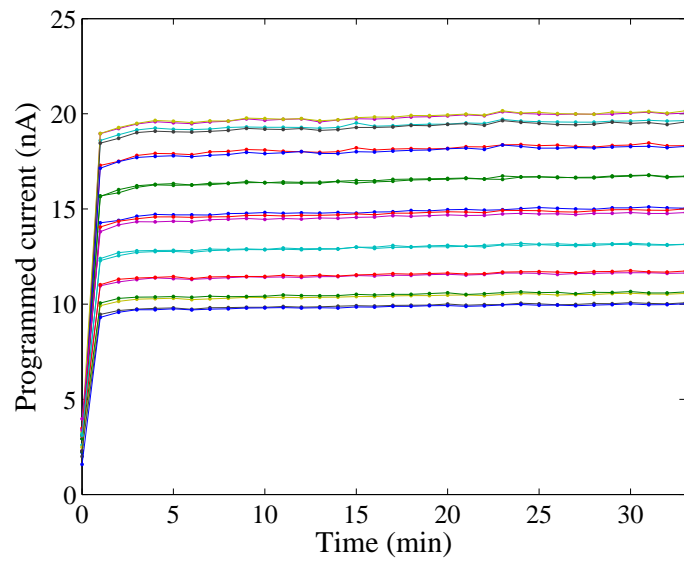
Figure 5.28: Calibration of the output current mirror.

Using the method, we programmed a sinusoidal filter $I_{pgm} = 15 + 5 \sin(\frac{\pi}{9}i)$, where $i = 0, 1, \dots, 18$. Fig. 5.29(a) shows the snap shot of the shape of the filter at different time points, showing the progression towards the final sinusoidal shape. Fig. 5.29(b) shows the time evolution of individual coefficient. Most of the programming is done in the first minute of the process. Fig. 5.30 shows the output current to target current ratio after the programming. The ratio ranges from 98.38% to 101.54%, with $\frac{\delta(ratio)}{\mu(ratio)} = \frac{0.89\%}{99.74\%} = 0.89\%$.

After the programming is done, we apply different input currents to test the linearity of the multiplication. Fig. 5.31 shows that the outputs maintain the *sin*



(a) Snap shot of the programming of a sine shape filter, \times : target current.



(b) Time evolution of the coefficients during programming a sine shape filter.

Figure 5.29: Program a sinusoidal filter.

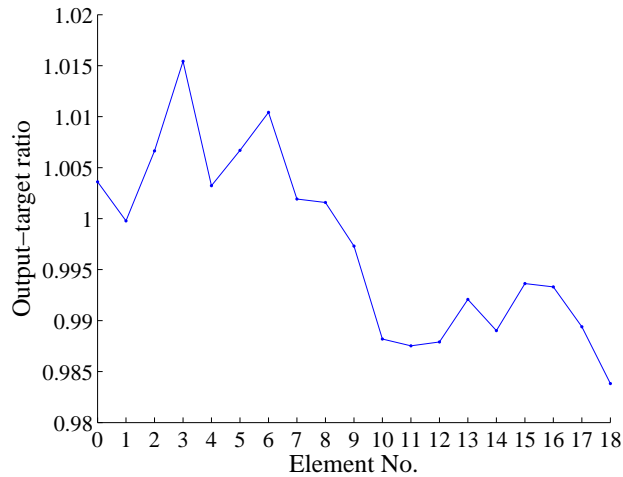
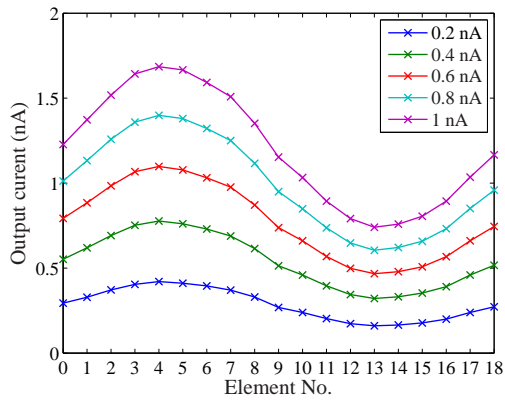


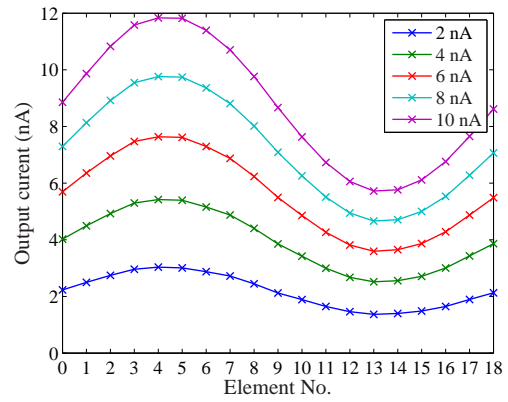
Figure 5.30: Output current to target current ratio after programming.

shape for the input currents from 0.2 nA to 40 nA. Fig. 5.32 shows the output current to input current ratio for inputs from 0.2 nA to 100 nA. It shows good linearity in the range of 20 nA to 100 nA.

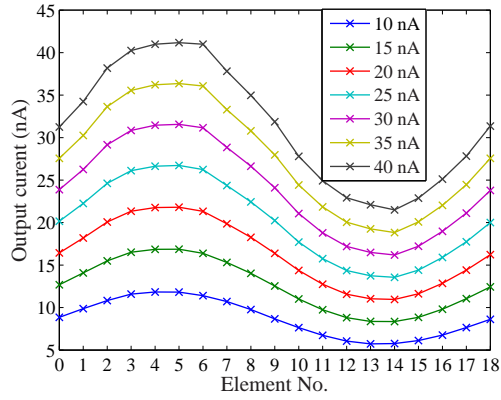
Next we tested an array of PCEs with input from an EMD array. We use a Lego set to provide the motion stimuli to the sensor to avoid the temporal aliasing from the monitor due to screen refreshing. A belt is driven by two DC motors, and it can move in different speeds and opposite directions depending on the voltage applied to the motors. When projected with light, the belt forms dark-bright gratings, with the spatial wavelength 0.8 cm. A lens with focal length of 6mm is used. The object to lens distance is 11 cm, thus giving the spatial frequency of the stimuli $f_s \approx \frac{1}{4.16^\circ}$. The DC motor has a good linearity, with 41.7 rpm/V. The belt moves 10 grates per motor rotation, thus giving the temporal frequency of the stimuli $f_t = 6.94 \text{ Hz/V}$. The velocity of the stimuli $v = \frac{f_t}{f_s} = 28.9^\circ/(s \cdot V)$. The setup is shown in the picture in Fig 5.33. We measured the response of the EMDs with the motor voltage from -7



(a)



(b)



(c)

Figure 5.31: Filtering a constant current input with the *sin* filter.

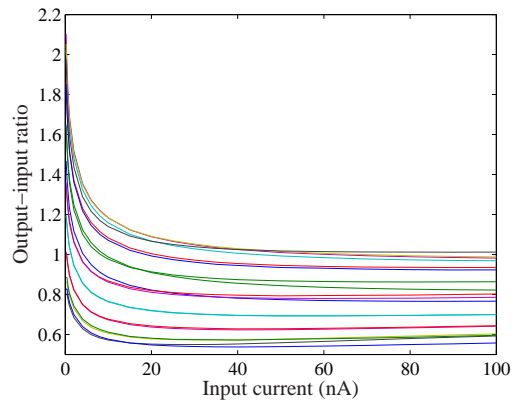


Figure 5.32: Output current to input current ratio as a function of the input current.

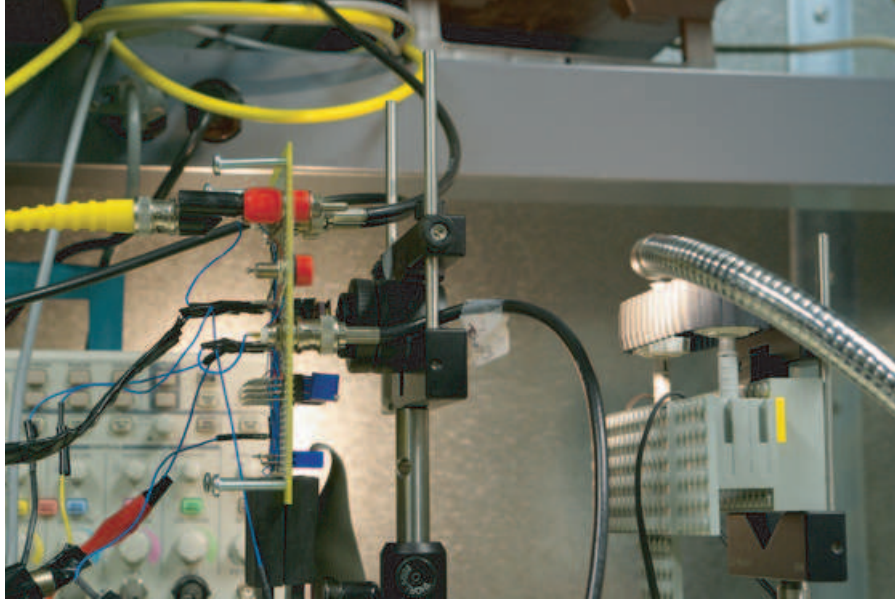
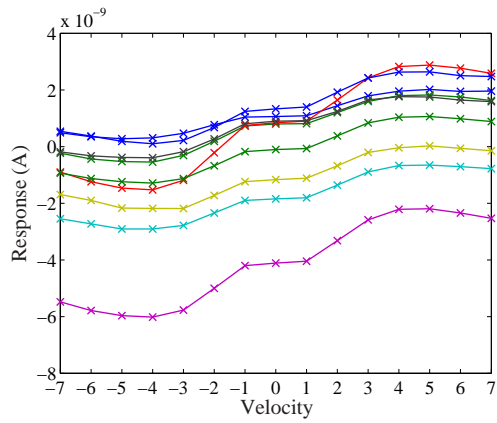


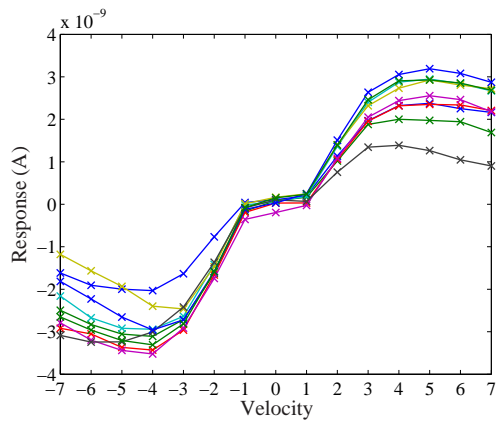
Figure 5.33: Photo of the flyeye test setup.

V to 7 V. Fig. 5.34(a) and Fig. 5.34(b) shows the velocity tuning curve before and after mismatch compensation. By tuning the bias of the low-pass filter, the peak of the velocity tuning can be changed. In Fig. 5.35, the averaged EMD velocity tuning curve is plotted. It shows that the higher the bias current, the smaller the time constant, thus the higher the peak velocity.

In the motion image sensor for the ground vehicle, four filters are needed: DC, $\sin(\cdot)$, $\cos(\cdot)$, $\cos(2\cdot)$. We simplify the filters to the two quadrant filters: 2 , $2 + \sin(\cdot)$, $2 \cos(\cdot)$, $2 + \cos(2\cdot)$. For the differential input from EMD, 8 first quadrant filters are needed in total. In the chip flyeye, we have 8 filters and 19 PCEs per filter. The size of the photodiode in an EMD is $45.5 \mu\text{m} \times 70 \mu\text{m}$, with intra-EMD distance $55.3 \mu\text{m}$. The size of an EMD is $112 \mu\text{m} \times 257.25 \mu\text{m}$ with inter-EMD distance $133 \mu\text{m}$. The pitch size of a PCE is $70 \mu\text{m} \times 74.2 \mu\text{m}$. We measured the initial PCE outputs when the chip was first powered up. We did the same measurement after we



(a) Before.



(b) After.

Figure 5.34: Mismatch compensation of the EMD velocity tuning, 1 velocity unit = $28.9^\circ/s$.

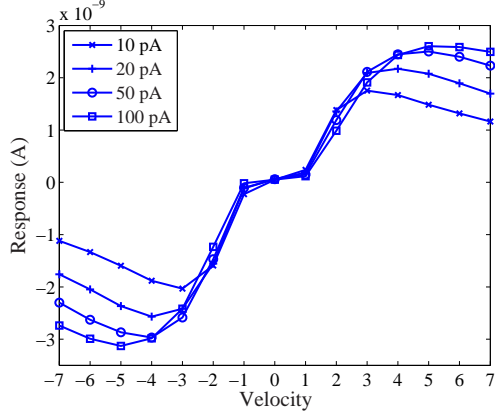
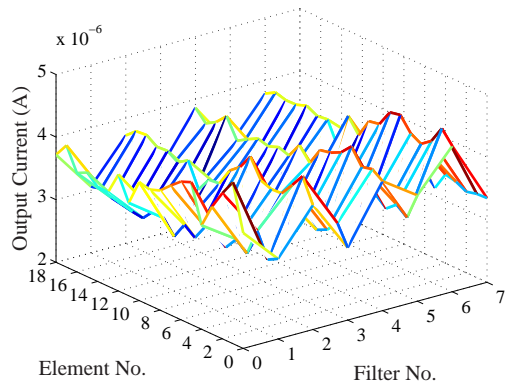


Figure 5.35: Averaging velocity tuning curve of EMDs for different bias current of low-pass filter, 1 velocity unit = $28.9^\circ/s$.

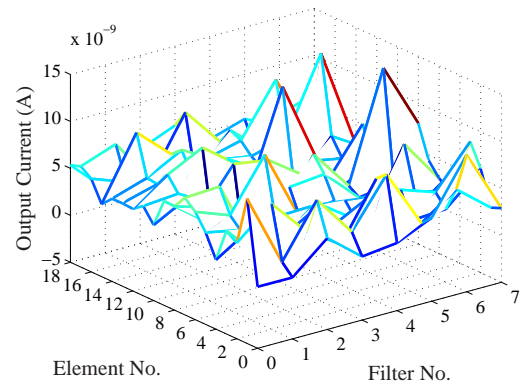
used tunnelling to set the floating gate voltage high enough to have small channel current. We then performed mismatch compensation using $I_{target} = 30$ nA. After mismatch compensation is done, V_{dd2} is set back to 5.0 V from 5.3 V. This shift of V_{dd2} reduces the PCE output. By lowering V_{tun} we can bring the current level back to the programmed level. We measured the PCE output at each stages. Fig. 5.36 shows the PCE output at different stages. And Table 5.1 shows the statistics of the measurement. Fig. 5.37 shows the velocity tuning curve of EMDs from one

Table 5.1: Statistics of the PCE output

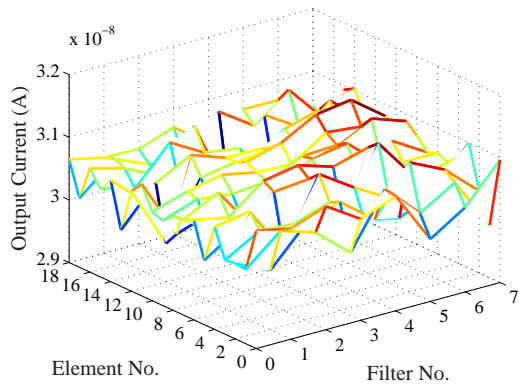
	max (nA)	min (nA)	μ (nA)	δ (nA)	$\frac{\delta}{\mu}$ (%)
Initial	4576	2789	3555	445	145.6
After tunnelling	14.7	0	4.7	2.5	8.2
After programming	31.2	29.6	30.5	0.32	1.0
After decreasing Vdd2	16.9	15.4	16.3	0.27	0.87
After tuning with Vtun	30.8	29.0	30.0	0.33	1.1



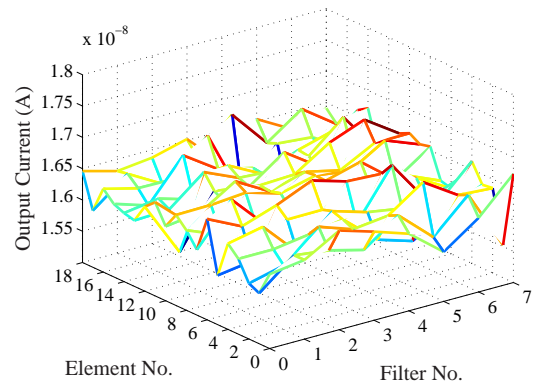
(a) Initial output



(b) Output after tunnelling

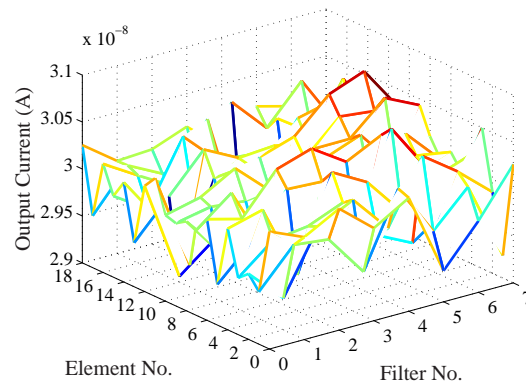


(c) Output after programming



(d) Output after programming with $V_{dd2} = 5.0$

V and $V_{tun} = 5$ V

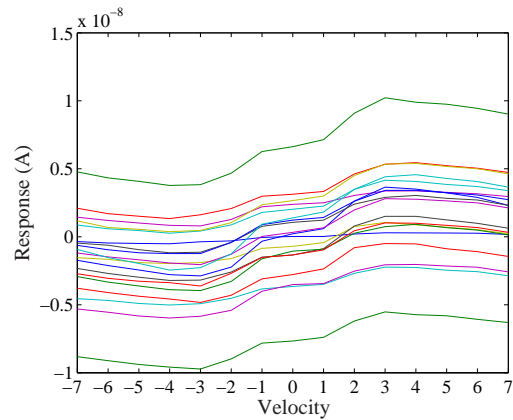


(e) Output after programming with $V_{dd2} = 5.0$

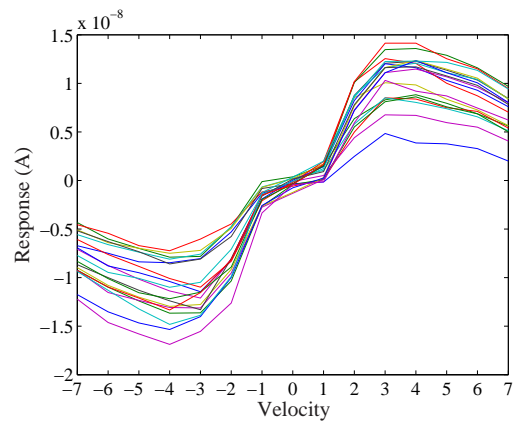
V and $V_{tun} = 3.48$ V

Figure 5.36: Distribution of output current from all PCEs from four filters.

filter before and after the mismatch compensation. To interface with the ground vehicle, the output current from the filter is converted to voltage by transimpedance amplifier. Fig. 5.38 shows the total EMD responses from four filters.



(a) Before.



(b) After.

Figure 5.37: Mismatch compensation of the EMD response in one filter, 1 velocity unit = $28.9^\circ/s$.

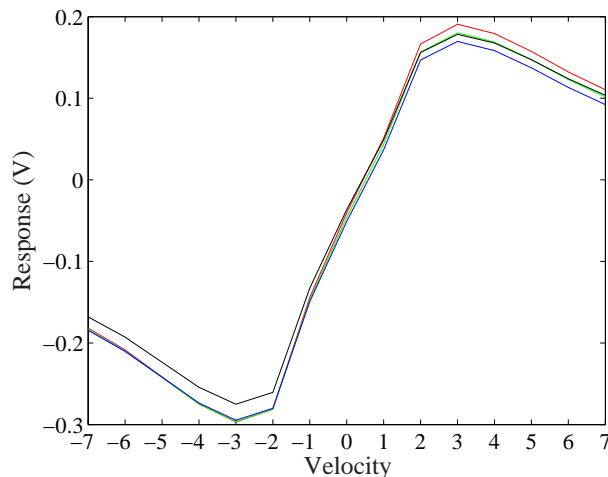


Figure 5.38: Total EMD responses after mismatch compensation from four filters, 1 velocity unit = $28.9^\circ/s$.

5.4.2 System Integration and Test

5.4.2.1 System Integration

We use a modified version of the vehicle WiRobot X80 from Dr Robot® [168] provided by the Autonomous Vehicle Laboratory in the Department of Aerospace Engineering. The vehicle is 15 inch in diameter with two 12 V motors driving 7 inch wheels. Fine wheel speed control is achieved by the sensing and controller module employing two 1200 count per wheel-cycle quadrature encoders. The vehicle has an on-board computer that provides the WiFi capability and computing power. We design a PCB board to interface with the vehicle. An off-shelf wide angle miniature lens is used. The lens has a fixed focal length of $f = 1.68$ mm. A custom made fixture is built to attach the lens to the chip 40-pin dual in-line package (DIP) in front of the sensor and to provide the right back focal length for the image to project

onto the sensor. On the board are also circuit elements to provide the biases to the sensor and to convert current outputs from the sensor to voltage signals. The board is mounted on the vehicle at the center (entrance pupil of the lens at the center of the vehicle), facing front. The voltage signals from the board are acquired by the data acquisition systems on the vehicle (NI PCI-6244) and processed to compute the command to drive the robot. The command is sent to the vehicle through the series port. Therefore, The computation up to the filtering is performed by the motion image sensor, and the computation and the communication of the command from the filter output is performed by the computer using Labview on the vehicle. At this stage, it is easier to interface the sensor with the vehicle through the computer. In a highly integrated system, dedicated hardware could be used to directly interface the sensor with the controller of the vehicle. Fig. 5.39 shows the picture of the sensor on the PCB board. Fig. 5.40 shows the picture of the integrated system. Fig. 5.41 shows the block diagram of the system.

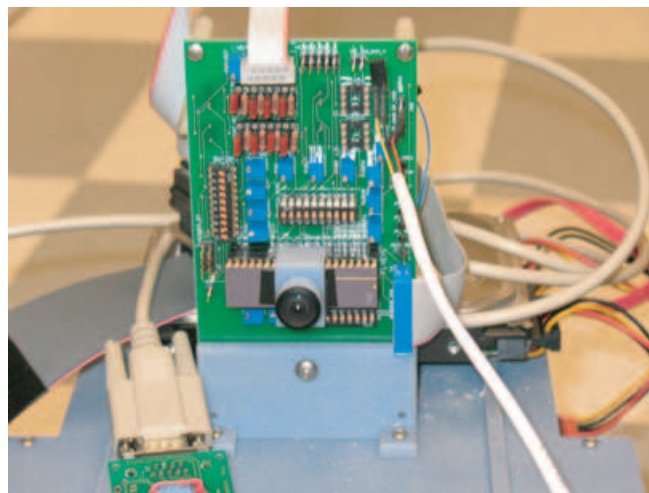


Figure 5.39: The sensor and the PCB board.

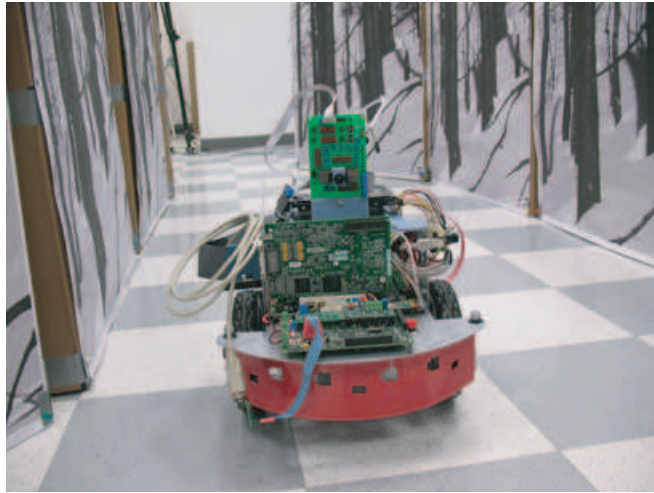


Figure 5.40: The integrated sensor and vehicle.

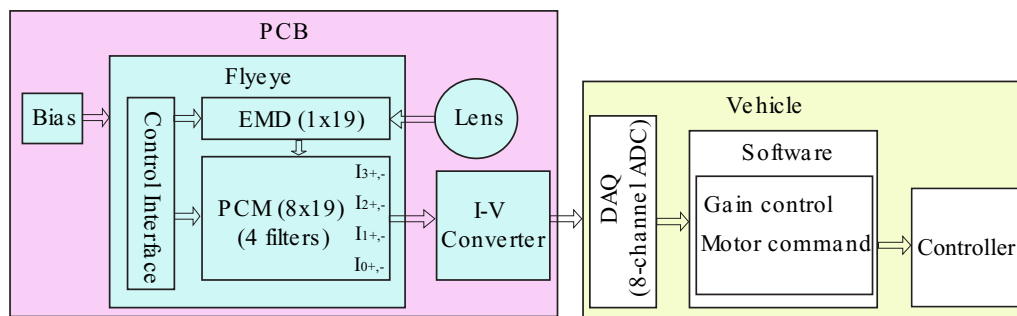


Figure 5.41: Block diagram of the system.

5.4.2.2 View Angle Characterization and Filter Programming

To program the spatial filter, the view angle of each EMD is needed. Since there is no image output from the sensor and motion response is the only output, we use the individual EMD response to find its view angle. Fig. 5.42 illustrates

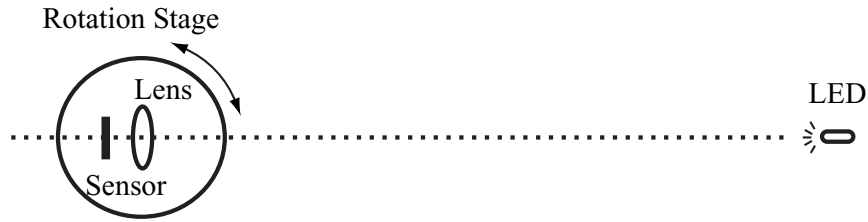


Figure 5.42: View angle calibration.

the experiment to calibration the EMD view angle. The sensor and the lens are on a rotation stage and are aligned with a LED light source on an optical rail. The LED is driven by a 10 Hz sinusoidal signal to generate sinusoidal optical signal. The light source is small enough that only one photodiode in one EMD can receive the light. To calibrate the view angle of each EMD, we rotate the stage and monitor the response of the tested EMD. When the LED projects to the left photodiode of the EMD, the EMD has large positive response, while when the LED projects to the right photodiode of the EMD, the EMD has large negative response. The view angle of the EMD is determined by the mean value of the angles corresponding to the large positive and negative responses. The measured view angle of the 19 EMDs is

$$\gamma = [44.83^\circ, 39.73^\circ, 34.77^\circ, 30.19^\circ, 25.58^\circ, 20.87^\circ, 16.49^\circ, 11.86^\circ, 7.47^\circ, 2.90^\circ, \\ -1.94^\circ, -6.24^\circ, -10.77^\circ, -15.46^\circ, -19.88^\circ, -24.72^\circ, -29.16^\circ, -33.27^\circ, -38.23^\circ]$$

The mean inter-EMD angle is 4.61° . The calculated mean inter-EMD angle is $2 \cdot \arctan\left(\frac{\Delta D}{2f}\right) \frac{180}{\pi} \approx 4.53^\circ$, where $\Delta D = 133 \mu\text{m}$ is the layout distance between EMDs. The total span of view angle is 83.06° .

We then programmed four filters $a_o = 2$, $a_1 = 2 \cos(\gamma)$, $a_2 = 2 + \cos(2\gamma)$, and $b_1 = 2 + \sin(\gamma)$. The target currents to program are $I_{target} = 12 \times 2 \text{ nA}$, $12 \times 2 \cos(\gamma) \text{ nA}$, $12 \times (2 + \cos(2\gamma)) \text{ nA}$, and $12 \times (2 + \sin(\gamma)) \text{ nA}$. Fig. 5.43 shows the programmed current I_{pgm} of four filters and the scaled I_{target} . The standard deviation and mean of the scalars are

$$\delta = [0.0059, 0.0059, 0.0037, 0.0068]$$

$$\mu = [0.9800, 0.9789, 0.9771, 0.9617]$$

Since the variations of the scalars are small, they only cause little distortion of the filter shape.

5.4.2.3 Open Loop Test

We measured the filter outputs while the vehicle rotating in a open loop setup. Fig. 5.44 shows the mean filter outputs. Since all EMD units experience the same optic flow for a constant rotation velocity, the filter outputs are the EMD velocity tuning curves scaled by the summation of the coefficients in the filters. The offsets at the zero velocity are caused by the current voltage converters and can be easily removed. We calculate the ratio between the filter outputs after removing the offsets. Table 5.2 shows the ratio from the measurement and the ideal case. They agree very well.

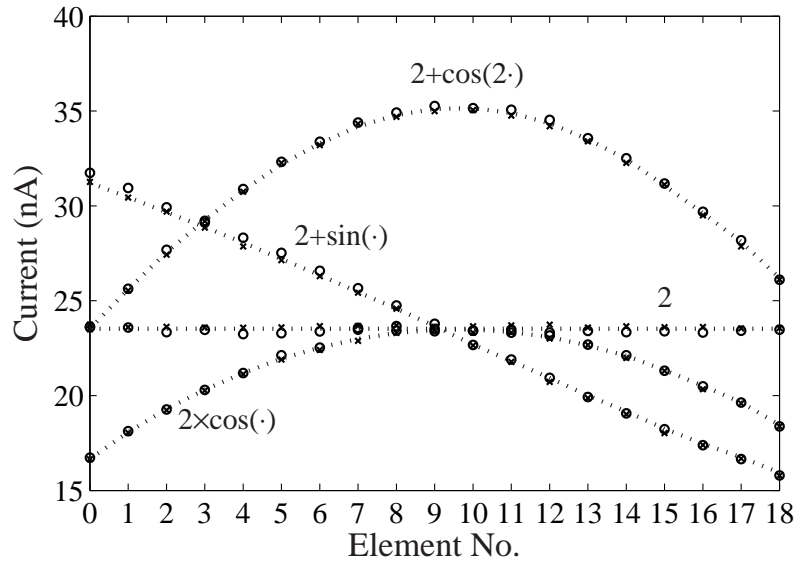


Figure 5.43: Program four filters: $a_0 = 2$, $a_1 = 2 \cos(\cdot)$, $a_2 = 2 + \cos(2\cdot)$, $b_1 = 2 + \sin(\cdot)$. \circ : programmed current from filter taking minus part of the input, \times : programmed current from filter taking plus part of the input. Dotted line: scaled ideal filter current.

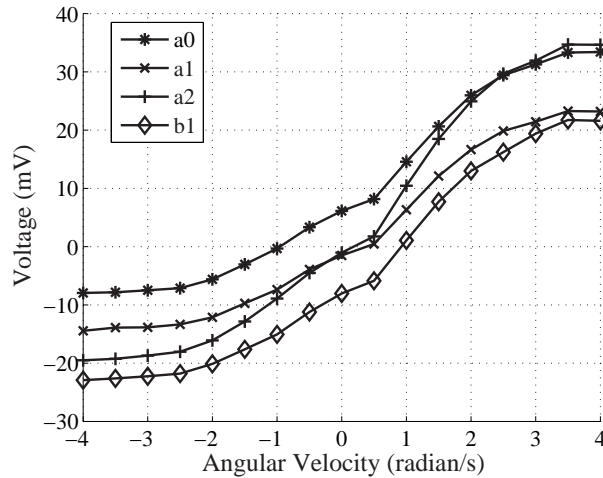


Figure 5.44: Mean filter outputs at constant angular velocities, positive angular velocity: counter clockwise, negative angular velocity: clockwise.

Table 5.2: Ratio of filter output during rotation

	Experimental	Ideal
$\frac{a_1}{a_0}$	0.9106	0.9059
$\frac{a_2}{a_0}$	1.3040	1.3279
$\frac{b_1}{a_0}$	1.0680	1.0228

5.4.2.4 Closed Loop Test

Since we can only control the wheel velocity of the robot rather than the torque, we use a kinematic model of the robot motion in the inertial frame [61]

$$\dot{v} = u_1 \tag{5.29}$$

$$\dot{y} = v \sin \theta \tag{5.30}$$

$$\dot{\theta} = u_2 \tag{5.31}$$

where u_1 and u_2 are the control input and v is the forward speed. The linearized state equation of $p = (v, y, \theta)$ around the centerline constant speed trajectory $p_0 = (v_0, 0, 0)$ is

$$\dot{p} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & v_0 \\ 0 & 0 & 0 \end{pmatrix} p + \begin{pmatrix} u_1 \\ 0 \\ u_2 \end{pmatrix} \tag{5.32}$$

The linearized WFI output from the sensor are

$$\begin{pmatrix} z_{a_1} \\ z_{a_2} \\ z_{b_1} \end{pmatrix} = \begin{pmatrix} 0 & Z_{2 \times 2} \\ \frac{1}{6a}(8 + \cos 3\beta - 9 \cos \beta) & 0 \end{pmatrix} \begin{pmatrix} v \\ y \\ \theta \end{pmatrix} \quad (5.33)$$

$$Z_{2 \times 2} = \begin{pmatrix} \frac{-v_0}{6a^2}(-3 \sin \beta + \sin 3\beta) & -\frac{v_0}{6a}(\cos 3\beta + 3 \cos \beta - 4) \\ \frac{v_0}{8a^2}(-\sin 4\beta + 4 \sin 2\beta - 4\beta) & \frac{v_0}{8a}(1 - \cos 4\beta) \end{pmatrix} \quad (5.34)$$

where a is the half width of the robot and $\beta = 41.5^\circ$ is the half view angle from the sensor. The speed regulation and obstacle avoidance are decoupled. In the closed loop experiment, the vehicle maintains constant speed by its internal controller and WFI output is used for obstacle avoidance. We choose $u_2 = K_1 z_{a_1} + K_2 z_{a_2}$ so that the linearized closed loop dynamics of (y, θ) is

$$\begin{pmatrix} \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} 0 & v_0 \\ \frac{v_0}{a^2}(0.1940K_1 + 0.1039K_2) & \frac{v_0}{a}(0.3866K_1 + 0.2463K_2) \end{pmatrix} \begin{pmatrix} y \\ \theta \end{pmatrix} \quad (5.35)$$

The characteristic function is

$$s^2 - \frac{v_0}{a}(0.3866K_1 + 0.2463K_2)s - \frac{v_0^2}{a^2}(0.1940K_1 + 0.1039K_2) = 0 \quad (5.36)$$

For stability we require $0.3866K_1 + 0.2463K_2 < 0$ and $0.1940K_1 + 0.1039K_2 < 0$, so that $K_1 < -0.6371K_2$ and $K_1 < -0.5356K_2$. We choose $K_1 = -2K_2$, the damping ratio is greater than one when $K_2 > 4.1$.

We conducted the closed loop experiments in various tunnel settings. The tunnel is formed by boxes with the inner wall covered by the printout of natural scenes. Fig. 5.40 shows the vehicle in a tunnel setup. The vehicle is set for constant velocity and filter outputs a_0 , a_1 , and a_2 are used to compute $u = K_1 a_1 + K_2(a_2 - a_0)$

for rotation control to navigate through the tunnel according to the framework shown in chapter 4. We use $v = 12$ inch/s (30.48 cm/s), $K_1 = 55$, and $K_2 = -55$. The filter outputs are sampled at 1 kHz and the average of every 50 samples are used to generate the command to the vehicle at 20 Hz. The experiment was conducted in the normal room lighting condition (~ 225 Lux). The position of the vehicle was tracked by the motion capture system Vicon®. The vehicle managed to navigate through the tunnel settings as shown in Fig. 5.45. The tunnel is about 12 feet (365.76 cm) long and 4 feet (122 cm) wide. The navigation is achieved using very limited information from only one horizontal line of optical input within 83.06° view angle.

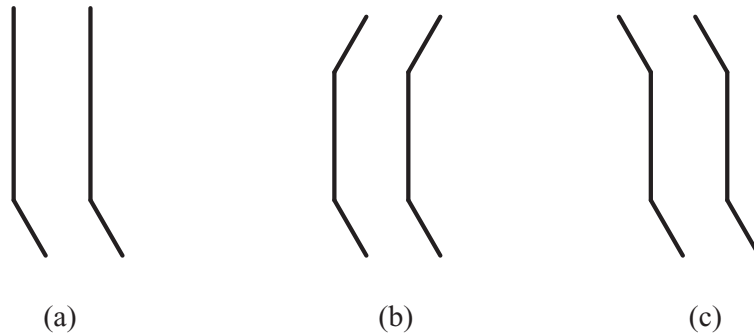


Figure 5.45: Three tunnel settings for closed loop experiments.

For the tunnel setup (a) and (b) we conducted multiple runs for three initial conditions: (1) initial y offset $y = -24.1$ cm, (2) initial θ offset $\theta = 21^\circ$, (3) no initial offset. Except for the tunnel setup (b) and initial condition (3) where 2 runs failed out of 22 runs, 20 consecutive successful runs were obtained for all other combinations excluding apparent system glitches such as the windows system freeze, low battery, etc. We plot the individual trajectories from the successful runs and

the mean trajectory. We also compute the mean standard deviation δ_y of the y coordinates along the trajectories. Fig. 5.46, 5.47, and 5.48 show the trajectories of the vehicle navigating through the tunnel setup (b) for 20 runs at three initial conditions.

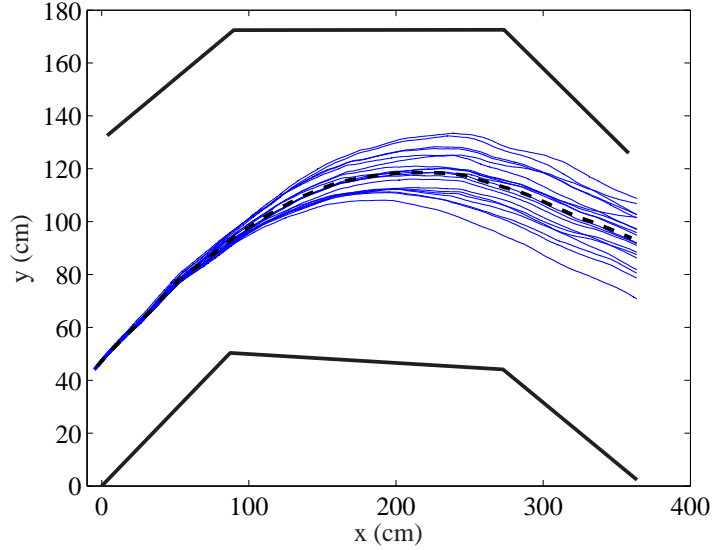


Figure 5.46: Trajectories of the vehicle navigating through the tunnel for 20 runs with initial offset $y = -24.1$ cm, dashed line: mean trajectory, $\delta_y = 6.6$ cm.

The measured total power consumption by the PCB is 57 mW. It is mostly consumed by the operational amplifiers (opamps) on the board. There are 16 opamps, 8 for voltage followers to buffer the chip output for debugging or to provide voltage biases to the chip, and 8 for current voltage converters. Each opamp consumes about 3.5 mW and they consume 56 mW in total. Since we do not have the access to the current at the power pin of the chip, we estimate the chip power consumption here. The bias current of the Gilbert multiplier (I_{bm}) is the dominant current in EMDs, the power consumption of each EMD is about $V_{dd}I_{bm} = 100$ nW for $I_{bm} = 10$ nA.

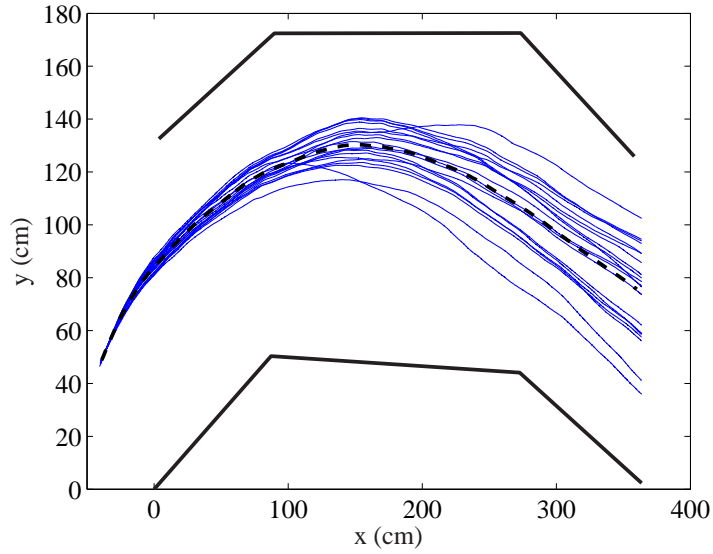


Figure 5.47: Trajectories of the vehicle navigating through the tunnel for 20 runs with initial offset $\theta = 21^\circ$, dashed line: mean trajectory, $\delta_y = 9.6$ cm.

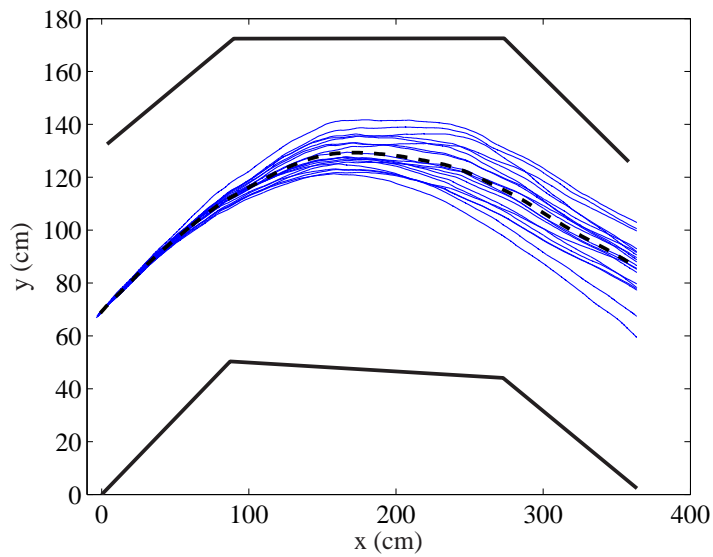


Figure 5.48: Trajectories of the vehicle navigating through the tunnel for 20 runs with no initial offset, dashed line: mean trajectory, $\delta_y = 6.8$ cm.

The current mirrors between EMDs and PCM consume the same amount of power as EMDs. The power consumption in each PCE is determined by the output current from EMD (I_{o+} or I_{o-}) and the programmed coefficient. Since I_{o+} and I_{o-} are scaled by the same coefficient F_c from the two-quadrant filters and $I_{o+} + I_{o-} = 2I_{bm}$, the power consumption by a pair of PCE is $4F_cV_{dd}I_{bm}$. Thus the total power consumption by the PCM is $38.82 \mu\text{W}$, and the total power consumption by the EMD and PCM is $42.6 \mu\text{W}$. The rest of the circuits on-chip are bias circuits and the control logic in static states. They consume much less power than the sensing and computational core of the sensor. So the total estimated power consumption of the sensor is about $42.6 \mu\text{W}$. We can reduce the board level power consumption by moving the bias circuits on-chip.

Chapter 6

Conclusions and Open Problems

This dissertation presented a body of work that roots from the appreciation of the elegance of the neural computation in biology and inspiration by the underlying principles. The goal is to find solutions to engineering challenges based on biological solutions from millions of years evolution.

In chapter 2, we demonstrated the first silicon stochastic synapse with short-term depression (STD). It shows the similar behavior as its biological counterpart and matches the theory and simulation of a STD circuit model based on a biological model. We make use of the intrinsic circuit noise which is often treated as adverse features and curtailed at the designer's best effort. As the technology pushes into nanometer regime, device becomes inevitably stochastic. New computational paradigm based on stochastic elements need to be investigated. Stochastic coding and computation utilized by biology might be one of the solutions. The demonstration of the silicon stochastic synapse with STD is just the beginning of the biologically inspired stochastic computation. As proposed in the dissertation, other forms of short term plasticity and long term plasticity need to be implemented and demonstrated with the silicon stochastic synapse. Different functionalities and applications of the dynamic stochastic synapse need to be explored. For example, stochastic synapse with STD could be used in coincidence detection to avoid the

influence of spike rate on the actual coincidence. Furthermore, stochastic synapses need to be incorporated with other existing VLSI spiking neural systems. The impact of the randomness brought by the stochastic synapses on the network dynamics and learning behavior could be an interesting research direction.

To fully understand the circuit stochastic behavior, we proposed a method for stochastic circuit modeling and transient analysis using stochastic differential equations in chapter 3. We applied the method to explain the stochastic behavior of the stochastic synapse circuit. We further developed a small signal stochastic model for the circuit. The model shows us the role each device plays in the circuit's stochastic behavior so that we can predict and tune the circuit performance. Using the same method, we analyzed the stochastic behavior of a CMOS inverter operating at very low power supply from first principles. This analysis provides insights about the expected behavior of the basic digital element affected by noise increasingly due to power supply scaling. This method could be potentially developed into a useful tool in circuit simulators for transient analysis of noise effects. In our model, we limited the noise to white noise. It is worthwhile to extend the method to other types of noise.

From chapter 4, we moved to a higher level of functions of neural systems, sensorimotor integration. Aiming at developing highly integrated sensor actuator systems for micro air vehicle, we investigated optic flow wide field integration (WFI) based navigation inspired by the fly. We demonstrated in simulation that the fly inspired elementary motion detector (EMD) is a potential candidate for optic flow estimation for WFI based navigation. We extended previous theoretical result to a

general case of limited view angles.

In chapter 5, we designed and demonstrated a motion image sensor with on-chip optic flow estimation, adaptation, and programmable spatial filtering for WFI based navigation. We proposed a novel approach to compensate the fabrication mismatch among the EMDs across the field of view. We further demonstrated that the same structure can be used to program the analog spatial filter and perform filtering operation. This is the first single chip solution to WFI based navigation where detailed spatial structure of the optic flow is used to extract motion cues. The method proposed here can also be applied to other sensory front-end where on-chip spatial processing is required and distortion from fabrication mismatch among sensor units has to be reduced. Although the sensor is geared for low power and light weight application such as the micro air vehicle, we first integrated the sensor with a ground vehicle for proof of concept. The vehicle could navigation through simple tunnel environments with limited information from only one horizontal line of optical input of height 2.4° and field of view angle 83.1° .

The computation involved in the WFI based navigation is rather simple. It is the perfect case where analog VLSI would excel than the conventional digital image processing approach. The sensing and computation in the analog VLSI sensor is performed on-the-fly in parallel without any storage cost and bulky equipment associated with the conventional approach. In addition the sensor consumes low power, with the current implementation consuming about $42.6 \mu\text{W}$. Altogether the VLSI motion image sensor is an ideal smart sensor to provide optic sensing capability to micro air vehicles or insectlike robots, where small size and low power

consumption are mandatory. Micro air vehicles and insectlike robots could have a wide range of applications beyond spying as initially proposed. Ad hoc mobile sensor network, search-and-rescue operation, hazardous environment exploration and monitoring, planetary exploration, building inspection are just a few potential applications [169]. Many of the tasks need the deployment of a large number of these devices, requiring low cost of the devices. Again analog VLSI sensors present the clear advantage. Insectlike robots have attracted increasing attention and interest recently and substantial progress has been made to manufacture robots in the size scale similar to insects [169]. It is foreseeable that the fly inspired sensor and control scheme demonstrated here would be a very efficient solution to the autonomous navigation for these types of robots in minuscule. The real-time continuous output from the sensor would enable fast response of the system. It may be the only approach by which the engineered system could achieve the similar agile behavior as its biological counterpart.

We took a quick approach to demonstrate the sensor and control scheme with an on-vehicle computer to interface the sensor with the vehicle controller. However, in a truly integrated system, the sensor output must be interfaced with the controller directly. The on-vehicle computer could be eliminated by making a dedicated PCB board where the sensor output is sampled and processed, and the command is computed, encoded and sent to the vehicle by an on-board micro-controller. However this post sensor computation is still in the flavor of digital computing. To fully appreciate the advantage of analog VLSI, specialized motors need to be developed so that they could be driven by the analog output from the sensor directly without

going to the digital domain. This could further miniaturize the system, reducing the complexity and power consumption while speeding up the system response.

The EMD output is an approximation of the optic flow with the ambiguity from the non-monotonic response to velocity. Removing the ambiguity can certainly improve the velocity range of the operation. This could be achieved by using the normalized response between two EMDs as explained in Chapter 4. The EMD output is affected by contrast, therefore very noisy. Reducing the sensitivity to contrast could smooth the response and improve the accuracy of the motion cue extracted. A new motion image sensor with the contrast adaptive EMDs proposed by Shoemaker and O'Carroll [143] is designed and fabricated. The performance could be compared with the one without contrast adaptation. Increasing the receptive field of the sensor can also improve the system performance. This could be done by using multiple sensors or specialized optics to achieve omnidirectional vision [170].

Low power and small scale also make the analog VLSI sensor a good candidate for neuroprosthesis application. The sensor could be inserted to the retina of a blind person and help him to navigate around. This could be done by either interfacing the sensor with the patient's vision system using micro stimulator to generate the motion sensation, or convey the direction generated from the sensor output to the patient through other channels, such as voice prompt.

The filter operation in the sensor is essentially a vector dot product. Similar operation has been used in many applications under different names, such as matched filter, correlation, and similarity evaluation. The filter output indicates how closely the input matches the filter pattern. Therefore our approach can also

be used in applications where pattern recognition needs to be done in low cost and low power, such as portable device for hazard detection, medical monitoring.

Analog approach has its own limitation. Its precision is low compared with digital processing. As Hahm [157] pointed out, analog circuit implementation is more power efficient for shorter and faster filters, while digital circuit is more power efficient for longer and slower filters. So our approach is best suitable for applications requiring fast response without high precision.

Appendix A

Adaptive Large Time Constant Filter

A.1 Introduction

Filters with large time constant are important for biomedical applications and neurally inspired circuits because of the intrinsic signal properties the filters process. The time constant is normally determined by the operating point of the circuits and the capacitance in the circuits. Here we focus on the first order GmC type filters. The time constant $\tau = \frac{C}{g_m}$, where g_m is the transconductance of the transconductance amplifier in the filter. To obtain large time constants, a large capacitor or a small g_m is required. The transconductance amplifier in the GmC filter can be operated in subthreshold with diode connected transistors on both branches (Fig. 5.4) to reduce g_m . To achieve $\tau = 30$ ms with the bias current to the amplifier $I_b = 50$ pA, a capacitor of ~ 160 pF is needed. It is still quite large. Large capacitors occupy substantial silicon areas. Varieties of techniques have been explored to obtain large time constant filter without increasing capacitance, such as current scaling [153], floating gate [171], and current division [154].

Sodagar [156] proposed a very simple modification to the conventional GmC filter by adding a second output branch without direct feedback as shown in Fig. A.1(a). For the filter implementation, the first output branch V_{o1} is connected to the inverting input V_{im} as the negative feedback. The steady state of the second

output branch V_{o2} is still controlled by the feedback at V_{o1} , but its time constant is boosted substantially. It can be viewed as the increase of the equivalent resistance at the output node. At the first output with direct feedback, $R_{eq} = \frac{1}{g_m}$, while at the second output without direct feedback, $R_{eq} = \frac{r_o}{2}$, where r_o is the output resistance of the transistors. At the subthreshold operation, $g_m = \frac{\kappa I_b}{2V_T}$ and $r_o = \frac{V_A}{I_b/2}$, where V_A is the early voltage. Thus, the time constant is amplified by $\frac{\kappa V_A}{2V_T}$ at V_{o2} . However, since there is no direct feedback at V_{o2} , V_{o2} is fully determined by the operation of $M10$ and $M7$. V_{o2} could deviate from V_{o1} significantly, leaving the linear operation region and approaching to the power rails, because of the mismatch between the two output branches. In this appendix, we use the programmability of the floating gate pFET to cancel the mismatch so that V_{o2} stays close to V_{o1} at steady states.

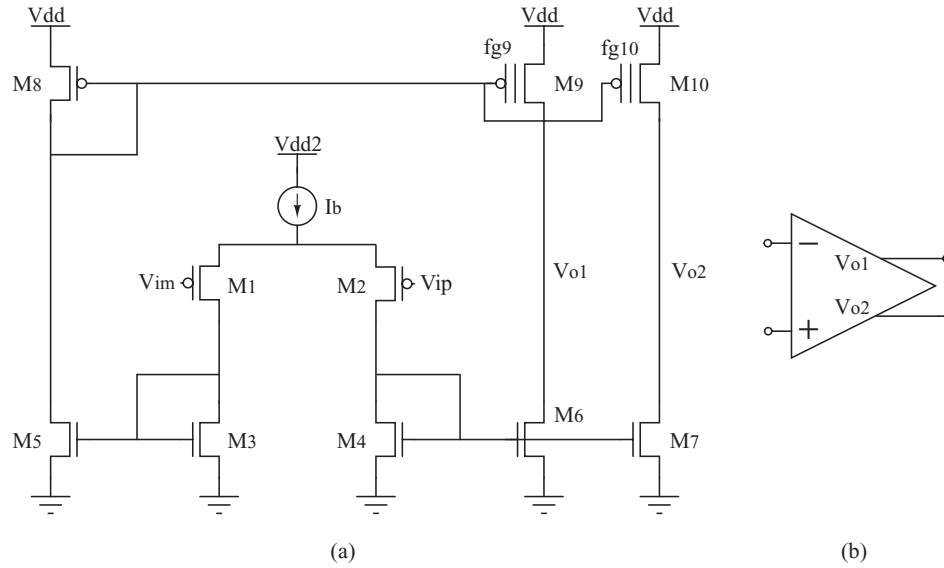


Figure A.1: The transconductance amplifier for the GmC filter with large time constant, (a) schematic, (b) symbol.

A.2 Basic Structure

As shown in Fig. A.1(a) we use floating gate pFETs $M9$ and $M10$ in the transconductance amplifier. We use injection current in $M10$ to compensate the mismatch between the two branches. Floating gate pFET is used for $M9$ as well for a better matching of the capacitance coupling with $M10$ from $M8$. For $M10$, when V_{dd} is high enough (>5.3 V) and V_{o2} is low enough due to the current imbalance between $M10$ and $M7$, current injection occurs in $M10$ so that $fg10$ decreases and its channel current increases. As a result, V_{o2} increases. The injection is stopped automatically when V_{o2} reaches the targeted voltage V_{o1} by a feedback control.

Fig. A.2 shows the block diagram of the circuits we use for the adaptive large time constant filter. The input to the non-inverting input of the transconductance amplifier is selected by the output from a high speed comparator comparing V_{o2} and a reference voltage V_{sw} . V_{o1} is connected to the inverting input for the direct negative feedback.

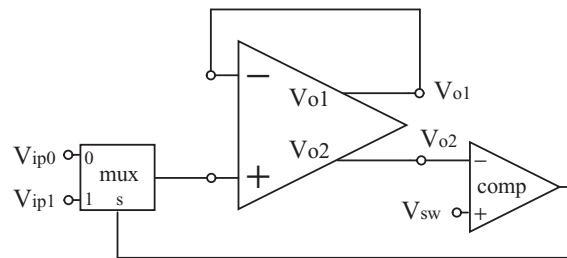


Figure A.2: The circuit block diagram of the adaptive large time constant filter.

A.3 Operation Configuration

Fig. A.3(a) shows the configuration during the programming mode. A DC voltage is applied at both V_{ip1} and V_{sw} . Another DC voltage much higher than V_{ip1} is applied at V_{ip0} . V_{dd} is set to 5.3 V. Initially, the floating node $fg10$ in Fig. A.1 is brought high enough through a tunnelling structure so that V_{o2} is close to ground. V_{ip1} is selected as the non-inverting input of the transconductance amplifier by the comparator output. The injection current at $M10$ causes V_{o2} to increase. Once it reaches V_{ip1} , V_{ip0} is selected as the non-inverting input by the comparator output. Since V_{ip0} is much higher than V_{ip1} , both V_{o1} and V_{o2} are driven high to V_{ip0} and the injection shuts off. At this stage, the mismatch between the two branches is cancelled, i.e., V_{o1} and V_{o2} are close for the same input. We can then configure the circuit as either a low-pass filter (LPF) (Fig. A.3(b)) or a high-pass filter (HPF) (Fig. A.3(c)), where V_i is the input and V_o is the output.

A.4 Simulation Results

We first simulate the conventional GmC filter, with $I_b = 100$ pA and $C = 100$ fF. The low-pass filter $f_{3dB} = 4.4$ kHz and the high-pass filter $f_{3dB} = 2.19$ kHz. The estimated $f_{3dB} = \frac{g_m}{2\pi C} = 2.23$ kHz.

We simulate the programming of the adaptive filter with an initial voltage offset at the floating gate: $fg9 = 4.6$ V, $fg10 = 4.8$ V, $V_{ip1} = 2$ V, and $V_{ip0} = 4$ V. Fig. A.4 shows the transient simulation result. Initially $V_{o2} = 0$ V and it approaches to V_{o1} while $fg10$ approaches to $fg9$. At 15.8 s the output of the comparator sel

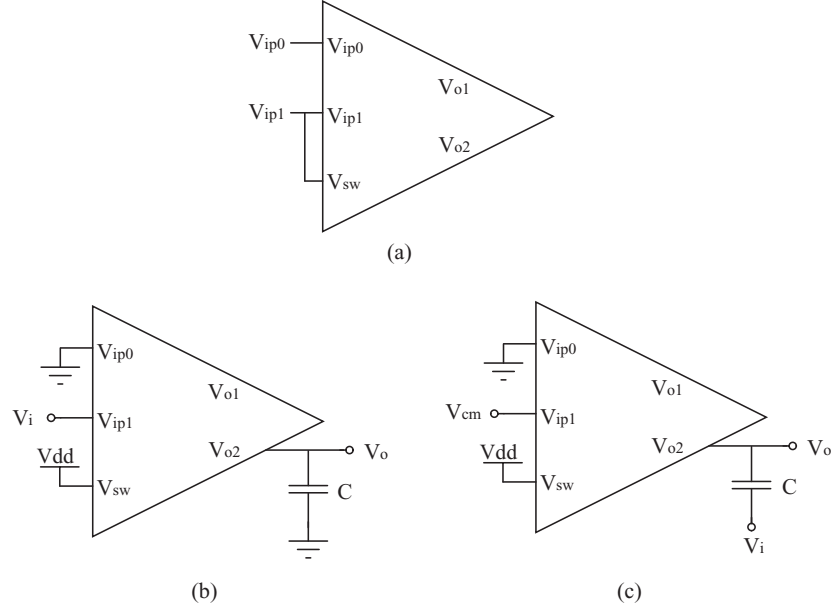


Figure A.3: The configuration of the adaptive large time constant filter, (a) programming, (b) low-pass filter, (c) high-pass filter.

switches; and both V_{o1} and V_{o2} jumps to 4 V and stay constant since there is no injection current any more. V_{dd} switches from 5.3 V to 5 V at 18 s and the circuit is ready for normal filter operation. The residue voltage difference between V_{o1} and V_{o2} is approximately 30 mV.

We then configure the circuit as a low-pass filter and a high-pass filter as shown in Fig. A.3(b) and (c), with $I_b = 100$ pA and $C = 100$ fF. f_{3dB} are 34.6 Hz and 31.1 Hz respectively, 127 times and 70 times smaller than the conventional filter. Table A.1 shows the cutoff frequency for different combinations of the bias current and the capacitance. The cutoff frequency is approximately proportional to the bias current and inversely proportional to the capacitance, according to the small signal model $f_{3dB} = \frac{I_b}{\pi V_A C}$. Fig. A.5 and A.6 show the sweep AC simulation of the capacitance for the LPF and HPF.

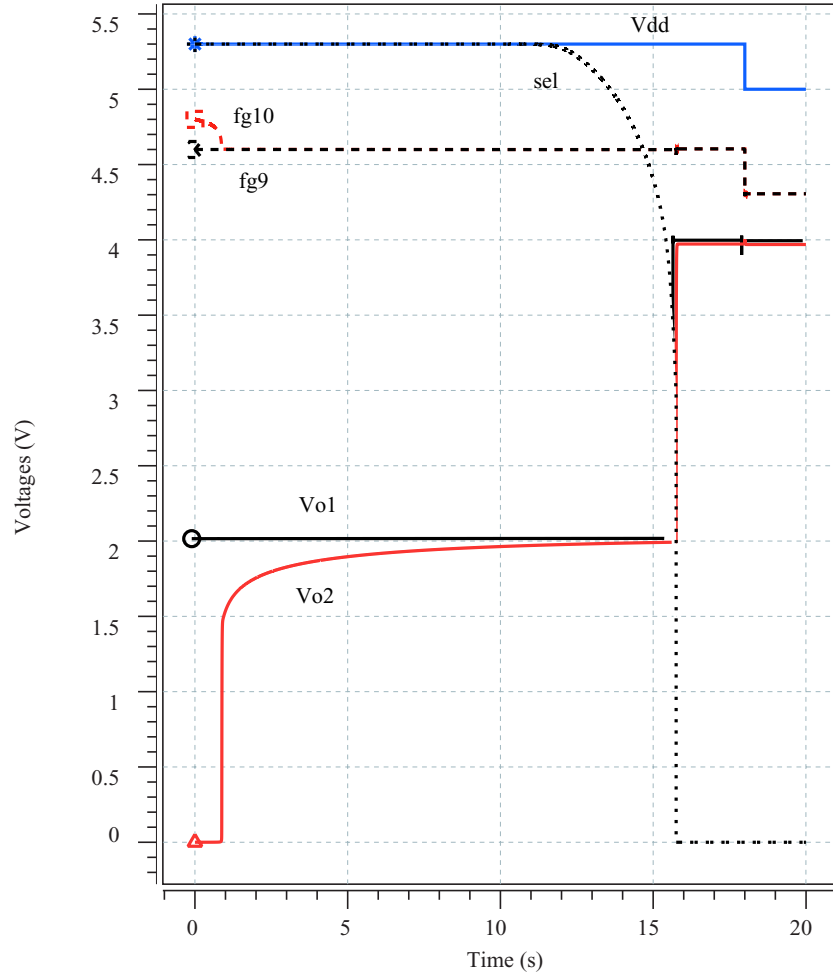


Figure A.4: Transient simulation of the programming of the adaptive filter.

Table A.1: Cutoff frequency

C	LPF		HPF	
	$I_b = 100 \text{ pA}$	$I_b = 1 \text{ nA}$	$I_b = 100 \text{ pA}$	$I_b = 1 \text{ nA}$
100 fF	34.6 Hz	292 Hz	31.3 Hz	278 Hz
1 pF	3.47 Hz	29.5 Hz	3.45 Hz	29.9 Hz

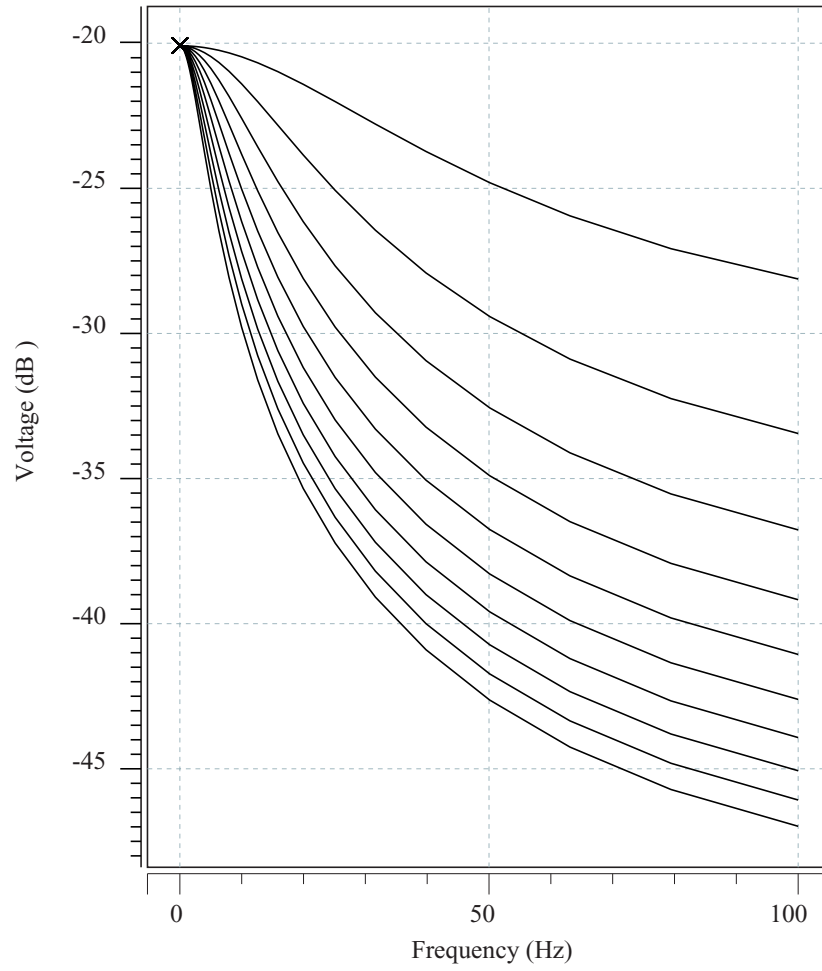


Figure A.5: Sweep AC analysis of the adaptive filter configured as a LPF with the capacitance from 100 fF to 1 pF in the increment of 100 fF.

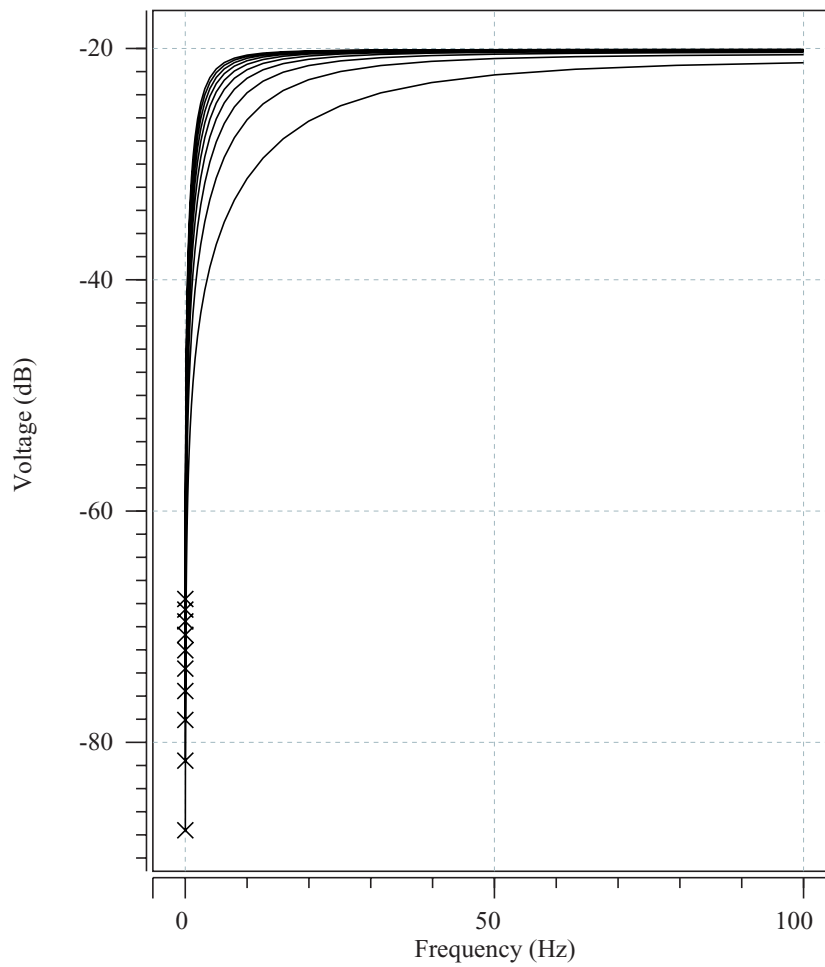


Figure A.6: Sweep AC analysis of the adaptive filter configured as a HPF with the capacitance from 100 fF to 1 pF in the increment of 100 fF.

Appendix B

Adaptive Floating Gate Pixel

Previously floating-gate MOSFET has been applied to CMOS imager to reduce the fixed pattern noise [32]. One of the drawback of the imager is its slow speed and low fill factor. In this appendix, we propose an improved structure which has few transistors and a unique feature that its frequency response can be enhanced by its output capacitance.

B.1 Floating Gate Pixel Structure

Fig. B.1 shows the previous floating gate pixel structure [32]. The photocurrent I_{ph} is first mirrored by $M1$ and $M3$, then converted to the voltage V_{ph} by the current conveyor. Signal row enables the current conveyor in the normal operation mode. The bias current I_c is provided through the column line col . $M3$ is a floating gate pFET and the charges on the floating gate V_{fg} cancel the mismatch among the pixels in the imager.

Fig. B.2 shows the proposed new adaptive floating gate pixel structure. The current mirror is removed and the photocurrent I_{ph} is directly converted to the voltage V_{ph} by the current conveyor formed by $M1$ and $M2$. There are now only three transistors in each pixel. Signal \overline{row} enables the current conveyor. The bias current I_c is provided through the column line col . $M1$ is a floating gate pFET

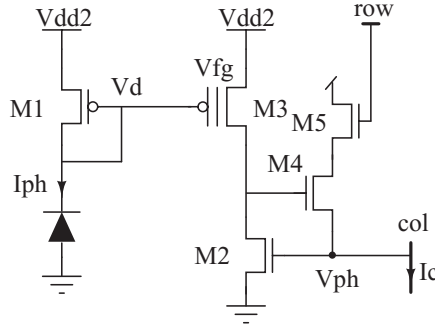


Figure B.1: Previous adaptive floating gate pixel structure.

and the charges on the floating gate V_{fg} cancel the mismatch among the pixels in the imager. During adaptation, uniform light is shined upon all pixels. All \overline{row} are high to disable the current conveyors and all V_{ph} are connected to a fixed voltage V_{cm} . V_{cm} is high enough so that the channel current of $M1$ is smaller than I_{ph} in each pixel and I_{ph} pulls down V_d . With V_{dd2} high enough, the current injection occurs in $M1$ and the channel current of $M1$ increases. This process continues until the channel current is equal to the photocurrent. At this stage, mismatch is compensated because all pixels have the same output voltage V_{cm} at V_{ph} for the same input light.

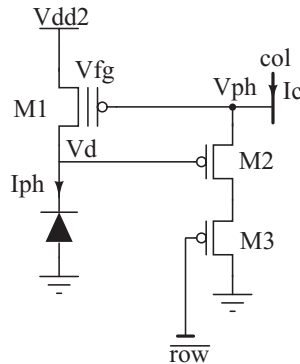


Figure B.2: New adaptive floating gate pixel structure.

B.2 Frequency Analysis

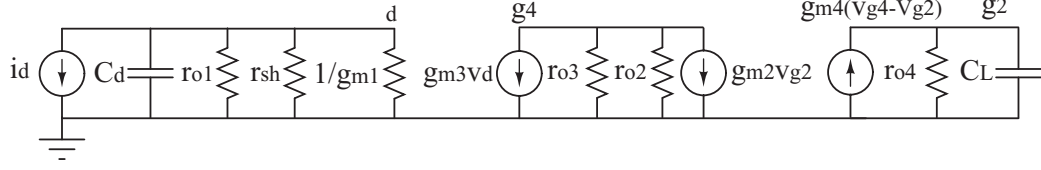


Figure B.3: Small signal model of the previous pixel structure.

We perform the small signal analysis of these two structures. In the analysis, the floating gate pFETs are replaced by the regular pFETs, and the row select transistors are ignored for simplification. Fig. B.3 shows the small signal model of the previous pixel, where C_d is the parasitic capacitance of the photodiode, r_{sh} is the shunting resistance of the photodiode, r_{o1} - r_{o4} are the output resistance of $M1$ - $M4$, C_L is the load capacitor.

$$v_{g2} = g_{m4}(v_{g4} - v_{g2}) \frac{1}{sC_L + \frac{1}{r_{o4}}} \quad (\text{B.1})$$

$$v_{g2} = \left(\frac{g_{m4}r_{o4}}{1 + g_{m4}r_{o4}} \right) \left(\frac{1}{1 + s \frac{r_{o4}C_L}{1 + g_{m4}r_{o4}}} \right) v_{g4} \quad (\text{B.2})$$

$$\approx \frac{1}{1 + s\tau'_2} v_{g4} \quad (\text{B.3})$$

$$\tau'_2 \approx \frac{C_L}{g_{m4}} \quad (\text{B.4})$$

$$v_{g4} = -(g_{m3}v_d + g_{m2}v_{g2})(r_{o2}||r_{o3}) \quad (\text{B.5})$$

$$= \left(\frac{i_d g_{m3}}{sC_d + g_{m1}} - g_{m2}v_{g2} \right) \frac{r_{o2}}{2} \quad (\text{B.6})$$

$$v_{g4} = \frac{i_d}{g_{m2}} \left(\frac{1}{1 + s\tau_1} \right) \left(\frac{1 + s\tau_2'}{1 + s\frac{\tau_2'}{1+g_{m2}r_{o2}/2}} \right) \quad (\text{B.7})$$

$$\tau_1 = \frac{C_d}{g_{m1}} \quad (\text{B.8})$$

$$v_{g2} = \frac{i_d}{g_{m2}} \left(\frac{1}{1 + s\tau_1} \right) \left(\frac{1}{1 + s\frac{\tau_2'}{1+g_{m2}r_{o2}/2}} \right) \quad (\text{B.9})$$

$$= \frac{i_d}{g_{m2}} \left(\frac{1}{1 + s\tau_1} \right) \left(\frac{1}{1 + s\tau_2''} \right) \quad (\text{B.10})$$

$$\tau_2'' = \frac{\tau_2'}{1 + g_{m2}\frac{r_{o2}}{2}} \quad (\text{B.11})$$

Normally $\tau_1 \gg \tau_2''$, so the pole determined by τ_1 is dominant for v_{g2} , v_{g2} is the small signal of the pixel output V_{ph} . The pixel frequency response is limited at the first stage by the photocurrent and the parasitic capacitance of the photodiode.

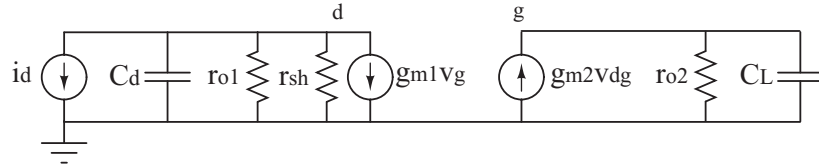


Figure B.4: Small signal model of the new pixel structure.

Fig. B.4 shows the small signal model of the new pixel.

$$v_d = -(i_d + g_{m1}v_g) \frac{1}{sC_d + \frac{1}{r_{o1}||r_{sh}}} \quad (\text{B.12})$$

$$\approx -(i_d + g_{m1}v_g) \frac{r_{o1}}{1 + sr_{o1}C_d} \quad (\text{B.13})$$

$$= -(i_d + g_{m1}v_g) \frac{r_{o1}}{1 + s\tau_1} \quad (\text{B.14})$$

$$\tau_1 = r_{o1}C_d \quad (\text{B.15})$$

assuming $r_{sh} \gg r_{o1}$.

$$v_g = g_{m2}(v_d - v_g) \frac{r_{o2}}{1 + sr_{o2}C_L} \quad (\text{B.16})$$

$$= g_{m2}(v_d - v_g) \frac{r_{o2}}{1 + s\tau_2} \quad (\text{B.17})$$

$$v_g = \left(\frac{g_{m2}r_{o2}}{1 + g_{m2}r_{o2}} \right) \left(\frac{1}{1 + s \frac{\tau_2}{1 + g_{m2}r_{o2}}} v_d \right) \quad (\text{B.18})$$

$$\approx \frac{1}{1 + s\tau'_2} v_d \quad (\text{B.19})$$

$$\tau_2 = r_{o2}C_L \quad (\text{B.20})$$

$$\tau'_2 = \frac{\tau_2}{1 + g_{m2}r_{o2}} \approx \frac{C_L}{g_{m2}} \ll \tau_2 \quad (\text{B.21})$$

Solving the equation, we obtain

$$v_d = \left(-\frac{i_d}{g_{m1}} \right) \left(\frac{g_{m1}r_{o1}}{1 + g_{m1}r_{o1}} \right) \left(\frac{1 + s\tau'_2}{1 + s \frac{\tau_1 + \tau'_2}{1 + g_{m1}r_{o1}} + s^2 \frac{\tau_1\tau'_2}{1 + g_{m1}r_{o1}}} \right) \quad (\text{B.22})$$

$$\approx \left(-\frac{i_d}{g_{m1}} \right) \left(\frac{1 + s\tau'_2}{1 + s\tau'_1 + s^2\tau'_1\tau'_2} \right) \quad (\text{B.23})$$

$$v_g = \left(-\frac{i_d}{g_{m1}} \right) \left(\frac{g_{m1}r_{o1}}{1 + g_{m1}r_{o1}} \right) \left(\frac{1}{1 + s \frac{\tau_1 + \tau'_2}{1 + g_{m1}r_{o1}} + s^2 \frac{\tau_1\tau'_2}{1 + g_{m1}r_{o1}}} \right) \quad (\text{B.24})$$

$$\approx \left(-\frac{i_d}{g_{m1}} \right) \left(\frac{1}{1 + s\tau'_1 + s^2\tau'_1\tau'_2} \right) \quad (\text{B.25})$$

$$\tau'_1 = \frac{\tau_1}{1 + g_{m1}r_{o1}} \approx \frac{C_d}{g_{m1}} \quad (\text{B.26})$$

v_g is the small signal of the pixel output V_{ph} . Similar to the previous case, τ'_1 is determined by the photocurrent and the parasitic capacitance of the photodiode. However, the first pole of v_g is determined by both τ'_1 and τ'_2 . The frequency response can be enhanced by adjusting τ'_2 . When $\tau'_2 \ll \tau'_1$, the dominant pole is determined by τ'_1 . As τ'_2 increases, the two poles approach each other so that the time constant from the dominant pole decreases. When $\tau'_2 = \frac{\tau'_1}{4}$, the two poles meet at $-\frac{2}{\tau'_1}$ and the pixel has the fast frequency response without overshoot. Thus the bandwidth

can be doubled with appropriate τ'_2 compared with the previous pixel. As τ'_2 keeps increasing, the two poles split, their real values stay at $-\frac{2}{\tau_1}$. The imaginary part of the poles causes over shoot in the response.

We perform AC analysis of both circuits. For $C_d = 100$ fF, $I_c = 1 \mu\text{A}$, $I_{ph} = 500$ pA, when $C_L < 5$ pF, both circuits have $f_{3dB} = 14.3$ kHz. For $C_L = 5, 10,$ and 25 pF, $f_{3dB} = 14.3, 14.5,$ and 15.5 kHz for the previous pixel, showing little influence from C_L , while $f_{3dB} = 15, 16.1,$ and 19.2 kHz for the new pixel.

In summary, the proposed adaptive floating gate pixel has two less transistors than the previous structure and the frequency response can be adjusted at the output stage with doubled bandwidth at the best.

Bibliography

- [1] C. Mead, *Analog VLSI and Neural Systems*. New York: Addison-Wesley, 1989.
- [2] J. S. Humbert, R. M. Murray, and M. H. Dickinson, "A control-oriented analysis of bio-inspired visuomotor convergence," in *Proc. IEEE Conf. on Decision and Control*, Seville, Spain, 2005, pp. 245–250, Dec. 12 - Dec. 15.
- [3] R. I. Bahar, J. Mundy, and J. Chen, "A probabilistic-based design methodology for nanoscale computation," in *Proc. International Conference on Computer-Aided Design*, San Jose, USA, Nov. 2003, pp. 480–486.
- [4] H. Chen, P. Fleury, and A. Murray, "Unsupervised probabilistic neural computation in mixed-mode vlsi," in *Smart adaptive systems on silicon*, M. Valle, Ed. Springer, 2004, pp. 825–837.
- [5] K. V. Palem, "Energy aware computing through probabilistic switching: A study of limits," *IEEE Trans. Comput.*, vol. 54, no. 9, pp. 1123–1137, Sept. 2005.
- [6] K. Nepal, R. Bahar, J. Mundy, W. Patterson, and A. Zaslavsky, "Designing logic circuits for probabilistic computation in the presence of noise," in *Proc. 42nd Design Automation Conference*, 2005, pp. 485–490.
- [7] N. H. Hamid, A. F. Murray, D. Laurenson, S. Roy, and B. Chen, "Probabilistic computing with future deep submicrometer devices: a modelling approach," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 88, 2005, pp. 2510–2513.
- [8] B. Akgul, L. Chakrapani, P. Korkmaz, and K. Palem, "Probabilistic CMOS technology: A survey and future directions," in *Proc. IFIP International Conference on VLSI*, Oct. 2006, pp. 1–6.
- [9] E. S. Fortune and G. J. Rose, "Short-term synaptic plasticity as a temporal filter," *Trends Neurosci.*, vol. 24, pp. 381–385, July 2001.
- [10] L. A. Grande and W. J. Spain, "Synaptic depression as a timing device," *Physiology*, vol. 20, no. 3, pp. 201–210, 2005.
- [11] L. F. Abbott, J. A. Varela, K. Sen, and S. B. Nelson, "Synaptic depression and cortical gain control," *Science*, vol. 275, pp. 220–224, 1997.
- [12] F. S. Chance, S. B. Nelson, and L. F. Abbott, "Synaptic depression and the temporal response characteristics of V1 cells," *J. Neurosci.*, vol. 18, no. 12, pp. 4785–4799, 1998.
- [13] F. Nadim and Y. Manor, "The role of short-term synaptic dynamics in motor control," *Curr. Opin. Neurobiol.*, vol. 10, pp. 683–690, Dec. 2000.

- [14] W. Maass and A. M. Zador, “Dynamic stochastic synapses as computational units,” *Neural Comput.*, vol. 11, no. 4, pp. 903–917, 1999.
- [15] H. S. Seung, “Learning in spiking neural networks by reinforcement of stochastic synaptic transmission,” *Neuron*, vol. 40, pp. 1063–1073, Dec. 2003.
- [16] W. B. Levy and R. A. Baxter, “Energy-efficient neuronal computation via quantal synaptic failures,” *J. Neurosci.*, vol. 22, no. 11, pp. 4746–4755.
- [17] S. Schreiber, C. K. Machens, A. V. M. Herz, and S. B. Laughlin, “Energy-efficient coding with discrete stochastic events,” *Neural Comput.*, vol. 14, no. 6, 2002.
- [18] M. S. Goldman, P. Maldonado, and L. F. Abbott, “Redundancy reduction and sustained firing with stochastic depressing synapses,” *J. Neurosci.*, vol. 22, no. 2, pp. 584–591, 2002.
- [19] M. S. Goldman, “Enhancement of information transmission efficiency by synaptic failures,” *Neural Comput.*, vol. 16, no. 6, pp. 1137–1162, 2004.
- [20] H. Markram, J. Lubke, M. Frotscher, and B. Sakmann, “Regulation of synaptic efficacy by coincidence of postsynaptic apss and epsps,” *Science*, vol. 275, no. 5297, pp. 213–215, 1997.
- [21] Y. Dan and M. M. Poo, “Spike timing-dependent plasticity of neural circuits,” *Neuron*, vol. 44, pp. 23–30, Sept. 2004.
- [22] W. Senn, H. Markram, and M. Tsodyks, “An algorithm for modifying neurotransmitter release probability based on pre- and postsynaptic spike timing,” *Neural Computation*, vol. 13, pp. 35–67, 2000.
- [23] A. Pavasović, A. G. Andreou, and C. R. Westgate, “Characterization of sub-threshold MOS mismatch in transistor for VLSI systems,” *J. VLSI Signal Processing*, vol. 8, pp. 75–85, 1994.
- [24] T. Serrano-Gotarredona and B. Linares-Barranco, “Systematic width-and-length dependent cmos transistor mismatch characterization and simulation,” *Analog Integrated Circuits and Signal Processing*, vol. 21, pp. 271–296, 1999.
- [25] D. Boning and S. Nassif, “Models of process variations in device and interconnect,” in *Design of High-Performance Microprocessor Circuits*, A. Chandrakasan, W. J. Bowhill, and F. Fox, Eds. Wiley, 2000, pp. 98–116.
- [26] P. R. Kinget, “Device mismatch and tradeoffs in the design of analog circuits,” *IEEE J. Solid-State Circuits*, vol. 40, no. 6, pp. 1212–1224, 2005.
- [27] P. Hasler, B. A. Minch, and C. Diorio, “Adaptive circuits using pFET floating-gate devices,” in *Proc. Advanced Research in VLSI*, 1999, pp. 215–229.

- [28] P. Hasler and T. Lande, "Overview of floating-gate devices, circuits, and systems," *IEEE Trans. Circuits Syst. II*, vol. 48, no. 1, pp. 1–3, January 2001.
- [29] P. Hasler, B. Minch, and C. Diorio, "An autozeroing floating-gate amplifier," *IEEE Trans. Circuits Syst. II*, vol. 48, no. 1, pp. 74–82, January 2001.
- [30] M. Cohen and G. Cauwenberghs, "Floating-gate adaptation for focal plane online nonuniformity correction," *IEEE Trans. Circuits Syst. II*, vol. 48, pp. 83–89, 2001.
- [31] Y. L. Wong, M. H. Cohen, and P. A. Abshire, "A floating-gate comparator with automatic offset adaptation for 10-bit data conversion," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 7, pp. 1316–1326, July 2005.
- [32] E. L. Wong, M. H. Cohen, and P. A. Abshire, "A 128x128 floating gate imager with self-adapting fixed pattern noise reduction," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 5, May 2005, pp. 5314–5317.
- [33] P. Hasler, C. Diorio, B. A. Minch, and C. Mead, "Single transistor learning synapses with long term storage," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 3, 1995, pp. 1660–1663.
- [34] C. Diorio, P. Hasler, B. A. Minch, and C. Mead, "A single-transistor silicon synapse," *IEEE Trans. Electron Devices*, vol. 43, pp. 1972–1980, Nov. 1996.
- [35] —, "A floating-gate MOS learning array with locally computed weight updates," *IEEE Trans. Electron Devices*, vol. 44, pp. 2281–2289, Dec. 1997.
- [36] R. R. Harrison, J. A. Bragg, P. Hasler, B. A. Minch, and S. P. DeWeerth, "A CMOS programmable analog memory-cell array using floating-gate circuits," *IEEE Trans. Circuits Syst. II*, vol. 48, no. 1, pp. 4–11, Jan. 2001.
- [37] S. Shah and S. Collins, "A temperature independent trimmable current source," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 1, 2002, pp. I713–I716.
- [38] A. Bandyopadhyay, J. Lee, R. Robucci, and P. Hasler, "MATIA: A programmable 80 μ W/frame CMOS block matrix transformation imager architecture," *IEEE J. Solid-State Circuits*, vol. 41, pp. 663–672, Mar. 2006.
- [39] A. Bandyopadhyay, P. Hasler, and D. Anderson, "A CMOS floating-gate matrix transform imager," *IEEE Sensors J.*, vol. 5, no. 3, pp. 455–462, June 2005.
- [40] D. Hsu, M. Figueroa, and C. Diorio, "Competitive learning with floating-gate circuits," *IEEE Trans. Neural Networks*, vol. 13, no. 3, pp. 732–744, May 2002.
- [41] P. Hasler and J. Dugger, "An analog floating-gate node for supervised learning," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 5, pp. 834–845, May 2005.

- [42] P. R. Gray, P. J. Hurst, S. H. Lewis, and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits*. John Wiley & Sons, 2001.
- [43] R. J. Baker, *CMOS Circuit Design, Layout, and Simulation*. John Wiley & Sons, 2005.
- [44] P. Bolcato and R. Poujois, “A new approach for noise simulation in transient analysis,” in *Proc. IEEE Int. Symp. Circuits Systems*, 1992, pp. 887–890.
- [45] Y. Dong and A. Opal, “Time-domain thermal noise simulation of switched capacitor circuits and delta-sigma modulators,” *IEEE Trans. Computer-Aided Design*, vol. 19, no. 4, pp. 473–481, 2000.
- [46] N. Franceschini, J. M. Pichon, and C. Blanes, “From insect vision to robot vision,” *Phil. Trans. R. Soc. Lond. B*, vol. 337, pp. 283–294, 1992.
- [47] R. C. Hardie and P. Raghu, “Visual transduction in *Drosophila*,” *Nature*, vol. 413, pp. 186–193, Sept. 2001.
- [48] A. Borst and J. Haag, “Neural networks in the cockpit of the fly,” *J. Comp. Physiol. A*, vol. 188, pp. 419–437, 2002.
- [49] M. A. Frye and M. H. Dickinson, “Closing the loop between neurobiology and flight behavior in *Drosophila*,” *Current Opinion in Neurobiology*, vol. 14, pp. 1–8, 2004.
- [50] G. Indiveri and R. Douglas, “ROBOTIC VISION: neuromorphic vision sensors,” *Science*, vol. 288, no. 5469, pp. 1189–1190.
- [51] R. R. Harrison and C. Koch, “A robust analog VLSI motion sensor based on the visual system of the fly,” *Autonomous Robots*, vol. 7.
- [52] M. V. Srinivasan, M. Poteser, and K. Kral, “Motion detection in insect orientation and navigation,” *Vision Res.*, vol. 39, pp. 2749–2766, 1999.
- [53] M. V. Srinivasan and S. Zhang, “Visual navigation in flying insects,” in *Neuronal Processing of Optic Flow*, M. Lappe, Ed. Cambridge, MA: Academic Press, 2000, pp. 67–92.
- [54] M. Egelhaaf and R. Kern, “Vision in flying insects,” *Curr. Opin. Neurobiol.*, vol. 12, pp. 699–706, 2002.
- [55] M. Egelhaaf, R. Kern, H. G. Krapp, *et al.*, “Neural encoding of behaviourally relevant visual-motion information in the fly,” *Trends Neurosci.*, vol. 25, no. 2, pp. 96–102, 2002.
- [56] M. V. Srinivasan and S. Zhang, “Visual motor computation in insects,” *Annu. Rev. Neurosci.*, vol. 27, pp. 679–696, 2004.

- [57] H. G. Krapp, “Neuronal matched filters for optic flow processing in flying insects,” in *Neuronal Processing of Optic Flow*, M. Lappe, Ed. Cambridge, MA: Academic Press, 2000, pp. 67–91.
- [58] R. Wehner, “Matched filters - neuronal models of the external world,” *J. Comp. Physiol. A*, vol. 161, pp. 511–531, 1987.
- [59] J. S. Humbert, *Bio-inspired Visuomotor Convergence in Navigation and Flight Control Systems*. Ph.D. thesis, California Institute of Technology, 2006.
- [60] J. S. Humbert, R. M. Murray, and M. H. Dickinson, “Sensorimotor convergence in visual navigation and flight control systems,” in *Proc. 16th IFAC World Congress*, Praha, Czech Republic, 2005.
- [61] J. S. Humbert, A. Hyslop, and M. Chinn, “Experimental validation of wide-field integration methods for autonomous navigation,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 2144–2149, Oct. 29 - Nov. 2.
- [62] W. Reichardt, “Autocorrelation, a principle for the evaluation of sensory information by the central nervous system,” in *Sensory Communication*, A. Rosenblith, Ed. MIT Press, 1961, pp. 303–317.
- [63] M. Egelhaaf and W. Reichardt, “Dynamic response properties of movement detectors: theoretical analysis and electrophysiological investigation in the visual system of the fly,” *Biol. Cybern.*, vol. 56, pp. 69–87, 1987.
- [64] M. Egelhaaf and A. Borst, “Transient and steady-state response properties of movement detectors,” *J. Opt. Soc. Am. A*, vol. 6, no. 1, pp. 116–127, 1989.
- [65] M. Egelhaaf, A. Borst, and W. Reichardt, “Computational structure of a biological motion-detection system as revealed by local detector analysis in the fly’s nervous system,” *J. Opt. Soc. Am. A*, vol. 6, no. 7, pp. 1070–1087, 1989.
- [66] S.-C. Liu, “A neuromorphic aVLSI model of global motion processing in the fly,” *IEEE Trans. Circuits Syst. II*, vol. 47, no. 12, pp. 1458–1467, Dec. 2000.
- [67] R. R. Harrison and C. Koch, “A robust analog VLSI Reichardt motion sensor,” *Int. J. Analog Integr. Circuits Signal Process.*, vol. 24, pp. 213–229, 2000.
- [68] C. M. Higgins, V. Pant, and R. Deutschmann, “Analog VLSI implementation of spatio-temporal frequency tuned visual motion algorithms,” *IEEE Trans. Circuits Syst. I*, vol. 52, no. 3, pp. 489–502, Mar. 2005.
- [69] R. R. Harrison, “A biologically inspired analog IC for visual collision detection,” *IEEE Trans. Circuits Syst. I*, vol. 52, no. 11, pp. 2308–2318, Nov. 2005.

- [70] B. Webb, “Robots in invertebrate neuroscience,” *Nature*, vol. 417, pp. 359–363, 2002.
- [71] B. Webb, R. R. Harrison, and M. A. Willis, “Sensorimotor control of navigation in arthropod and artificial systems,” *Arthropod structure and development*, vol. 33, pp. 301–329, 2004.
- [72] D. Johnston and S. M.-S. Wu, *Foundations of Cellular Neurophysiology*. Cambridge, MA: MIT Press, 1997.
- [73] V. Matveev and X.-J. Wang, “Implications of all-or-none synaptic transmission and short-term depression beyond vesicle depletion: a computational study,” *J. Neurosci.*, vol. 20, pp. 1575–1588, 2000.
- [74] C. Koch, *Biophysics of Computation: Information Processing in Single Neurons*. New York, NY: Oxford University Press, 1999.
- [75] M. V. Tsodyks and H. Markram, “The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability,” *Proc. Natl. Acad. Sci. USA*, vol. 94, pp. 719–723, 1997.
- [76] P. Häfliger and M. Mahowald, “Spike based normalizing Hebbian learning in an analog VLSI artificial neuron,” *Int. J. Analog Integr. Circuits Signal Process.*, vol. 18, no. 2-3, pp. 133–139, 1999.
- [77] C. Rasche and R. H. R. Hahnloser, “Silicon synaptic depression,” *Biol. Cybern.*, vol. 84, pp. 57–62, 2001.
- [78] S.-C. Liu, “Analog VLSI circuits for short-term dynamic synapses,” *EURASIP Journal on Applied Signal Processing*, vol. 2003, pp. 620–628, 2003.
- [79] M. Boegerhausen, P. Suter, and S.-C. Liu, “Modeling short-term synaptic depression in silicon,” *Neural Computation*, vol. 15, pp. 331–348, 2003.
- [80] E. Chicca, G. Indiveri, and R. Douglas, “An adaptive silicon synapse,” in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 1, Bangkok, Thailand, May 2003, pp. 81–84.
- [81] A. Bofill, A. F. Murray, and D. P. Thompson, “Circuits for VLSI implementation of temporally asymmetric Hebbian learning,” in *Advances in Neural Information Processing Systems*, S. B. T. G. Dietterich and Z. Ghahramani, Eds. Cambridge, MA, USA: MIT Press, 2002.
- [82] G. Indiveri, E. Chicca, and R. Douglas, “A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity,” *IEEE Trans. Neural Networks*, vol. 17, pp. 211–221, 2006.
- [83] C. S. Petrie and J. A. Connelly, “A noise-based IC random number generator for applications in cryptography,” *IEEE Trans. Circuits Syst. I*, vol. 47, no. 5, pp. 615–621, May 2000.

- [84] D. H. Goldberg, G. Cauwenberghs, and A. G. Andreou, “Probabilistic synaptic weighting in a reconfigurable network of VLSI integrate-and-fire neurons,” *Neural Networks*, vol. 14, pp. 781–793, 2001.
- [85] S. Fusi, M. Annunziato, D. Badoni, A. Salamon, and D. J. Amit, “Spike driven synaptic plasticity: theory, simulation, VLSI implementation,” *Neural Computation*, vol. 12, pp. 2227–2258, 2000.
- [86] E. Chicca, D. Badoni, V. Dante, M. D’Andreagiovanni, G. Salina, L. Carota, S. Fusi, and P. Del Giudice, “A VLSI recurrent network of integrate-and-fire neurons connected by plastic synapses with long-term memory,” *IEEE Trans. Neural Networks*, vol. 14, pp. 1297–1307, 2003.
- [87] B. L. Nelson, *Stochastic Modeling: Analysis and Simulation*. McGraw-Hill, 1995.
- [88] B. D. Ripley, *Stochastic Simulation*. John Wiley Sons, 1987.
- [89] W. T. Holman, J. A. Connelly, and A. B. Dowlatabadi, “An integrated analog/digital random noise source,” *IEEE Trans. Circuits Syst. I*, vol. 44, no. 6, pp. 521–528, June 1997.
- [90] R. C. Fairfield, R. L. Mortenson, and K. B. Coulthart, “An LSI random number generator (RNG),” in *Proc. CRYPTO’84*, May 1985, pp. 203–230.
- [91] M. Degaldo-Restituto, F. Medeiro, and A. Rodriguez-Vázquez, “Nonlinear switched-current CMOS IC for random signal generation,” *Electronics Letters*, vol. 29, no. 25, pp. 2190–2191, Dec. 1993.
- [92] T. Stojanovski and L. Kocarev, “Chaos-based random number generators-part I: analysis,” *IEEE Trans. Circuits Syst. I*, vol. 48, no. 3, pp. 281–288, Mar. 2001.
- [93] T. Stojanovski, J. Pihl, and L. Kocarev, “Chaos-based random number generators-part II: practical realization,” *IEEE Trans. Circuits Syst. I*, vol. 48, no. 3, pp. 382–385, Mar. 2001.
- [94] M. J. Bellido, A. J. Acosta, M. Valencia, A. Barriga, and J. L. Huertas, “Simple binary random number generator,” *Electronics Letters*, vol. 28, no. 7, pp. 617–618, Mar. 1992.
- [95] P. D. Hortensius, R. D. McLeod, and H. C. Card, “Parallel random number generation for VLSI systems using cellular automata,” *IEEE Trans. Comput.*, vol. 38, no. 10, pp. 1466–1473, Oct. 1989.
- [96] I. M. Bland and G. M. Megson, “Systolic random number generation for genetic algorithm,” *Electronics Letters*, vol. 32, no. 12, pp. 1069–1070, June 1996.

- [97] P. Xu, Y. Wong, T. Horiuchi, and P. Abshire, "Compact floating-gate true random number generator," *Electronics Letters*, vol. 42, no. 23, pp. 1346–1347, Nov. 2006.
- [98] R. S. Zucker, "Short-term synaptic plasticity," *Ann. Rev. Neurosci.*, vol. 12, pp. 13–31, 1989.
- [99] C. F. Stevens and Y. Wang, "Facilitation and depression at single central synapses," *Neuron*, vol. 14, pp. 795–802, Apr. 1995.
- [100] P. Xu, T. K. Horiuchi, A. Sarje, and P. Abshire, "Stochastic synapse with short-term depression for silicon neurons," in *Proc. IEEE Biomedical Circuits and Systems Conference, 2007*, Montreal, Canada, Nov. 2007, pp. 99–102.
- [101] B. Schneier, *Applied Cryptography*. New York, NY: John Wiley Sons, 1996.
- [102] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, *A statistical test suite for random and pseudorandom number generator for cryptographic applications*. NIST special publication 800-22, 2001.
- [103] P. Xu, T. Horiuchi, and P. Abshire, "Stochastic model and simulation of a random number generator circuit," in *Proc. IEEE Int. Symp. Circuits Systems*, Seattle, WA, May 2008, pp. 2977–2980.
- [104] P. Xu and P. A. Abshire, "Stochastic behavior of a CMOS inverter," in *Proc. 14th IEEE International Conference on Electronics, Circuits and Systems*, Marrakech, Morocco, Dec. 2007, pp. 94–97.
- [105] E. Allen, *Modeling with Ito Stochastic Differential Equations*. Netherlands: Springer, 2004.
- [106] P. E. Kloeden, E. Platen, and H. Schurz, *Numerical Solution of SDE through Computer Experiments*. Heidelberg, German: Springer, 1994.
- [107] C. W. Gardiner, *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences*. Heidelberg, German: Springer, 2004.
- [108] R. Sarpeshkar, T. Delbrück, and C. A. Mead, "White noise in MOS transistors and resistors," *Circuits and Devices Magazine, IEEE*, vol. 9, pp. 23–29, Nov. 1993.
- [109] A. van der Ziel, *Noise in Solid State Devices and Circuits*. New York: Wiley, 1986.
- [110] R. Landauer, "Solid-state shot noise," *Physical Review B*, vol. 47, no. 24, pp. 16 427–16 432, 1993.

- [111] J. L. Wyatt, Jr., and G. J. Coram, “Nonlinear device noise models: Satisfying the thermodynamic requirements,” *IEEE J. Electron Devices*, vol. 46, no. 1, pp. 184–191, Jan. 1993.
- [112] C. C. Enz, F. Krummenacher, and E. A. Vittoz, “An analytical MOS transistor model valid in all regions of operation and dedicated to low-voltage and low-current applications,” *Analog Integrated Circuits and Signal Processing*, vol. 8, no. 1, pp. 83–114, 1995.
- [113] J. D. Meindl and J. A. Davis, “The fundamental limit on binary switching energy for terascale integration (tsi),” *IEEE J. Solid-State Circuits*, vol. 35, no. 10, pp. 1515–1516, Oct. 2000.
- [114] P. Abshire, *Sensory Information Processing under Physical Constraints*. Ph.D. thesis, Johns Hopkins University, 2002.
- [115] S. Cheemalavagu, P. Korkmaz, and K. V. Palem, “Ultra low-energy computing via probabilistic algorithms and devices: CMOS device primitives and the energy-probability relationship,” in *Proc. International Conference on Solid State Devices and Materials*, Tokyo, Japan, Sept. 2004, pp. 402–403.
- [116] S. R. Shaw, “Anatomy and physiology of identified non-spiking cells in the photoreceptor-lamina complex of the compound eye of insects, especially Diptera,” in *Neurons Without Impulses*, A. Roberts and B. M. H. Bush, Eds. Cambridge, UK: Cambridge University Press, 1981, pp. 61–109.
- [117] S. B. Laughlin, “Matching coding, circuits, cells, and molecules to signals: general principles of retinal design in the fly’s eye,” *Progress in Retinal and Eye Research*, vol. 13, pp. 165–196, 1994.
- [118] J. Douglass and N. Strausfeld, “Visual motion-detection circuits in flies: Parallel direction- and non- direction-sensitive pathways between the medulla and lobula plate,” *J. Neurosci.*, vol. 16, pp. 4551–4562, 1996.
- [119] M. A. Frye and M. H. Dickinson, “Fly flight: a model for the neural control of complex behavior,” *Neuron*, vol. 32, pp. 385–388, Nov. 2001.
- [120] R. Hengstenberg, H. Krapp, and B. Hengstenberg, “Visual sensation of self-motions in the blowfly *Calliphora*,” in *Biocybernetics of Vision: Integrative Mechanisms and Cognitive Processes*, C. Taddei-Ferretti, Ed., 1994.
- [121] W. Reichardt, “Movement perception in insects,” in *Processing of optical data by organisms and by machines*, W. Reichardt, Ed. New York: Academic Press, 1969, pp. 465–493.
- [122] A.-K. Warzecha and M. Egelhaaf, “Intrinsic properties of biological motion detectors prevent the optomotor control system from getting unstable,” *Phil. Trans. R. Soc. Lond. B*, vol. 351, pp. 1579–1591, 1996.

- [123] M. V. Srinivasan, M. Lehrer, W. H. Kirchner, and S. W. Zhang, “Range perception through apparent image speed in freely flying honeybees,” *Vis. Neurosci.*, vol. 6, pp. 519–535, 1991.
- [124] L. F. Tammero and M. H. Dickinson, “Collision avoidance and landing responses are mediated by separate pathways in the fruit fly, *Drosophila melanogaster*,” *J. Exp. Biol.*, vol. 205, pp. 2785–2798, 2002.
- [125] M. V. Srinivasan, S. W. Zhang, J. S. Chahl, E. Barth, and S. Venkatesh, “How honeybees make grazing landings on flat surfaces,” *Biol. Cybern.*, vol. 83, pp. 171–183, 2000.
- [126] M. V. Srinivasan, S. W. Zhang, and N. Bidwell, “Visually mediated odometry in honeybees,” *J. Exp. Biol.*, vol. 200, pp. 2513–2522, 1997.
- [127] C. Gilbert and N. J. Strausfeld, “The functional organization of male-specific visual neurons in flies,” *J. Comp. Physiol. A*, vol. 169, pp. 395–411, 1991.
- [128] M. F. Land and T. S. Collett, “Chasing behaviour of houseflies,” *J. Comp. Physiol.*, vol. 89, pp. 331–357, 1974.
- [129] N. Boeddeker, R. Kern, and M. Egelhaaf, “Chasing a dummy target: Smooth pursuit and velocity control in male blowflies,” *Proc. R. Soc. Lond. B Biol. Sci.*, vol. 270, pp. 393–399, 2003.
- [130] M. V. Srinivasan and M. Davey, “Strategies for active camouflage of motion,” *Proc. R. Soc. Lond. B Biol. Sci.*, vol. 259, pp. 19–25, 1995.
- [131] E. W. Justh and P. S. Krishnaprasad, “Steering laws for motion camouflage,” *Proc. R. Soc. Lond. A*, vol. 462, pp. 3629–3643, 2006.
- [132] R. Kern, C. Petereit, and M. Egelhaaf, “Neural processing of naturalistic optic flow,” *J. Neurosci.*, vol. 25, no. 2, pp. 96–102, 2002.
- [133] P. Reinagel, “How do visual neurons respond in the real world?” *Curr. Opin. Neurobiol.*, vol. 11, pp. 437–442, Aug. 2001.
- [134] D. W. Dong, “Spatiotemporal inseparability of natural images and visual sensitivities,” in *Motion Vision: Computational, Neural, and Ecological Constraints*, J. M. Zanker and J. Zeil, Eds. Springer, 2001, pp. 371–380.
- [135] N. Boeddeker, J. P. Lindemann, M. Egelhaaf, and J. Zail, “Responses of blowfly motion-sensitive neurons to reconstructed optic flow along outdoor flight paths,” *J. Comp. Physiol. A*, vol. 191, pp. 1143–1155, 2005.
- [136] J. P. Lindemann, R. Kern, J. H. Hateren, H. Ritter, and M. Egelhaaf, “On the computations analyzing natural optic flow: Quantitative model analysis of the blowfly motion vision pathway,” *J. Neurosci.*, vol. 25, no. 27, pp. 6435–6448, 2005.

- [137] E. H. Adelson and J. R. Bergen, “Spatiotemporal energy models for the perception of motion,” *J. Opt. Soc. Am. A*, vol. 2, pp. 284–299, Feb. 1985.
- [138] A. Borst, “Modeling fly motion vision,” in *Computational Neuroscience: A Comprehensive Approach*, J. Feng, Ed. CRC Press, 2003, pp. 397–429.
- [139] D. L. Ruderman and W. Bialek, “Statistics of natural images: Scaling in the woods,” *Physical Review Letters*, vol. 73, no. 6, pp. 814–817, 1994.
- [140] D. L. Ruderman, “The statistics of natural images,” *Network: Comput. Neural Syst.*, vol. 5, pp. 517–548, 1994.
- [141] R. O. Dror, D. C. O’Carroll, and S. B. Laughlin, “Accuracy of velocity estimation by Reichardt correlators,” *J. Opt. Soc. Am. A*, vol. 18, pp. 241–252, Feb. 2001.
- [142] P. A. Shoemaker, D. C. O’Carroll, and A. D. Straw, “Velocity constancy and models for wide-field visual motion detection in insects,” *Biol. Cybern.*, vol. 93, pp. 275–287, 2005.
- [143] P. A. Shoemaker and D. C. O’Carroll, “Insect-based visual motion detection with contrast adaptation,” in *Proc. SPIE*, vol. 5783, 2005, pp. 292–303.
- [144] G. A. Horridge and L. Marcelja, “On the existence of fast and slow directionally sensitive motion detector neurons in insects,” *Proc. R. Soc. Lond. B Biol. Sci.*, vol. 248, pp. 47–54, 1992.
- [145] M. J. Wainwright, O. Schwartz, and E. P. Simoncelli, “Natural image statistics and divisive normalization: Modeling nonlinearities and adaptation in cortical neurons,” in *Statistical Theories of the Brain*, R. Rao, B. Olshausen, and M. Lewicki, Eds. MIT Press, 2001.
- [146] R. C. Dorf and R. H. Bishop, *Modern Control Systems*. Addison-Wesley, 1995.
- [147] J.-J. E. Slotine and W. Li, *Applied Nonlinear Control*. Pearson Education, 2004.
- [148] G. L. Barrows, *Mixed-mode VLSI Optic Flow Sensors for Micro Air Vehicles*. Ph.D. thesis, University of Maryland, College Park, 1999.
- [149] G. Barrows, K. Miller, and B. Krantz, “Fusing neuromorphic motion detector outputs for robust optic flow measurement,” in *Proc. IEEE International Joint Conference on Neural Networks*, vol. 4, Washington DC, USA, 1999, pp. 2296–2301, July 10 - July 16.
- [150] R. A. Yotter and D. M. Wilson, “A review of photodetector for sensing light-emitting reporters in biological systems,” *IEEE Sensors J.*, vol. 3, no. 3, pp. 288–303, June 2003.

- [151] T. N. Swe and K. S. Yeo, "An accurate photodiode model for DC and high frequency SPICE circuit simulation," *Nanotech*, vol. 1, pp. 362–365, 2001.
- [152] E. Sánchez-Sinencio and J. Silva-Martínez, "CMOS transconductance amplifiers, architectures and active filters: a tutorial," in *IEE Proc. Circuits Devices Syst.*, vol. 147, no. 1, Feb. 2000, pp. 3–12.
- [153] M. Steyaert, P. Kinget, W. Sansen, and J. V. D. Spiegel, "Full integration of extremely large time constants in CMOS," *Electronics Letters*, vol. 27, no. 10, pp. 790–791, May 1991.
- [154] J. Silva-Martínez and S. Solís-Bustos, "Design considerations for high performance very low frequency filters," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 2, Orlando, FL, US, May 1999, pp. 648–651.
- [155] A. Veeravalli, E. Sánchez-Sinencio, and J. Silva-Martínez, "Transconductance amplifier structures with very small transconductances: a comparative design approach," *IEEE J. Solid-State Circuits*, vol. 37, no. 6, pp. 770–775, June 2002.
- [156] A. Sodagar, "Fully-integrated implementation of large time constant Gm-C integrators," *Electronics Letters*, vol. 43, no. 1, pp. 23–24, Jan. 2007.
- [157] M. Hahm, E. Friedman, and E. Titlebaum, "Analog vs. digital: a comparison of circuit implementations for low-power matched filters," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 4, Atlanta, GA, US, May 1996, pp. 280–283.
- [158] S.-C. Liu, J. Kramer, G. Indiveri, T. Delbrück, and R. Douglas, *Analog VLSI: Circuits and Principles*. Cambridge, MA: MIT Press, 2002.
- [159] M. Mahowald, *An Analog VLSI System for Stereoscopic Vision*. Boston, MA: Kluwer, 1994.
- [160] H. Kobayashi, J. L. White, and A. A. Abidi, "An active resistor network for Gaussian filtering of images," *IEEE J. Solid-State Circuits*, vol. 26, pp. 738–748, May 1991.
- [161] M. Choi and A. A. Abidi, "A 6-b 1.3-Gsample/s A/D converter in 0.35- μm CMOS," *IEEE J. Solid-State Circuits*, vol. 36, pp. 1847–1858, Dec. 2001.
- [162] A. G. Andreou and K. A. Boahen, "Translinear circuits in subthreshold MOS," *Analog Integrated Circuits and Signal Processing*, vol. 9, pp. 141–166, Mar. 1996.
- [163] B. Gilbert, "A monolithic microsystem for analog synthesis of trigonometric function and their inverses," *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 1179–1191, Dec. 1982.

- [164] T. S. Lande, H. Ranjbar, M. Ismail, and Y. Berg, "An analog floating-gate memory in a standard digital technology," in *Proc. MicroNeuro 96*, 1996, pp. 271–276.
- [165] G. Cauwenberghs, C. Neugebauer, and A. Yariv, "An adaptive CMOS matrix-vector multiplier for large scale analog hardware neural network applications," in *Proc. IEEE International Joint Conference on Neural Networks*, Seattle, US, July 1991, pp. 507–511.
- [166] T. Yamasaki and T. Shibata, "An analog similarity evaluation circuit featuring variable functional forms," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 2, Sydney, Australia, May 2001, pp. 561–564.
- [167] K. Rahimi, C. Diorio, C. Hernandez, and M. Brockhausen, "A simulation model for floating-gate MOS synapse transistors," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 2, Phoenix, AZ, US, May 2002, pp. 532–535.
- [168] [Online]. Available: http://www.drrobot.com/products_item.asp?itemNumber=X80
- [169] R. Wood, "Fly, robot fly," *Spectrum*, vol. 45, no. 3, pp. 25–29, 1996.
- [170] J. Gluckman and S. Nayar, "Ego-motion and omnidirectional cameras," in *Proc. IEEE Int. Conf. on Computer Vision*, Jan. 1998, pp. 999–1005.
- [171] P. Hasler, B. Minch, and C. Diorio, "An autozeroing floating-gate bandpass filter," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 1, 1998, pp. 131–134.