Implementation of Bayesian Fly Tracking Model using Analog Neuromorphic Circuits

Alin Thomas Tharakan, Dheeraj Bhaskar and Chetan Singh Thakur
Indian Institute of Science, Bengaluru, India
Email: {alintharakan, dheerajb, csthakur}@iisc.ac.in

***Abstract*—There is a growing body of evidence that suggests that the neurons in the brain calculate the posterior probability of states and events based on observations provided by the sensory neurons. Based on this hypothesis, a neuromorphic framework is proposed, where the sensory neurons of the dragonfly make noisy observations of the fruit fly and uses the underlying Hidden Markov Model (HMM) to track the fruit fly in two dimensional space. The dragonfly estimates the target position by solving the Bayesian recursive equations online. This work presents a novel approach for implementing probabilistic networks using sub-threshold analog neuromorphic circuits, with the ability to perform the computation in real-time. This framework will pave the way to build complex probabilistic algorithms based on HMMs for low power real-time applications.**

***Index Terms*—Bayesian Inference, Bayesian Tracking, Translinear Circuits, Current Feedback, Winner Take All**

I. INTRODUCTION

Bayesian systems find numerous applications in object tracking [1] - [3], source localization [4], video analytics [5] etc.. Also, there is a growing body of evidence that suggests that the neurons in the brain calculate the posterior probability of states and events based on observations provided by the sensory neurons [1]. These systems are usually implemented in software. But for large systems, a real time implementation using software model is not feasible. This mandates the need for hardware implementation and many existing works [6] - [8] have realized the same. Murray [7] has developed pulse based mixed signal neural circuits. Thakur et al. [8] [9] have implemented an FPGA based stochastic design.

The existing realizations, being primarily digital are power and area intensive and cannot be scaled to mimic larger biological systems. In this work, the aim is to implement a Bayesian Tracking model (proposed by Paulin et al. [1] and discussed in detail in Section II-A) using a set of reusable analog neuromorphic blocks. Such an implementation would help in reducing the power and area consumption, thereby better mimicking the biological systems. Since the mathematical framework of all Bayesian systems remain the same, these blocks can be used to implement other Bayesian systems as well, thereby, demonstrating the potential of building probabilistic algorithms in hardware using a single set of building blocks with the ability to perform these computations in real-time for low power applications.

The proposed Bayesian system [1] [10], describes how the neurons in the optic lobe of a dragonfly could infer the future location of a fruit fly based on the sensory spikes generated as the fruit fly passes in front of the dragonfly's retina. The tracker learns the transition model for target movement, and the observation model for the noisy sensors, and uses these to estimate the target position by solving the Bayesian recursive equation online.

II. ALGORITHMIC FRAMEWORK

An algorithmic framework similar to the one described in [8] is used. The main contribution of this work is to show the implementation using sub-threshold analog circuits for low power and low form factor applications.

A. Bayesian Fly Tracking

Bayesian Fly Tracking Model (BFTM) models the cognitive system of a dragonfly which tracks the fruit fly in the presence of background noise. The sensory neurons of the dragonfly fire spikes probabilistically, if there is a fruit fly or a distractor (noise). Some simplifying assumptions are made:

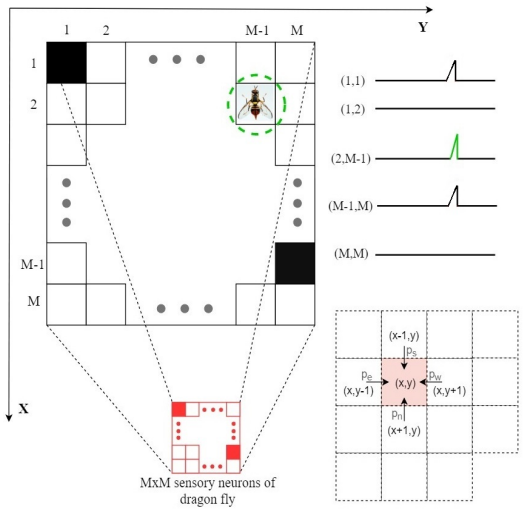

Fig. 1: **Dragonfly's field of vision**: There are *M*x*M* sensory neurons (SN) and the dragonfly's field of vision is divided into *M*x*M* regions. The corresponding SN may fire due to the presence of a distractor or fruit fly. For example, $SN_{(1,1)}$ and $SN_{(M-1,M)}$ (black spikes) fire due to presence of distractors whereas $SN_{(2,M-1)}$ (green spike) fires due to presence of fruit fly. There is no activity at the other sensory neurons. Also, the transition probabilities are shown as p_s, p_n, p_e and p_w.

1) There is only one fruit fly in the field of vision of the dragonfly, spanned by *M*x*M* sensory neurons as shown in Figure 1.
2) The motion of the fruit fly is restricted to two dimensions and it can move only one step in unit time (defined by the firing rate of the sensory neurons). One can associate probabilities to each of these movements - move west

(p_w), move north (p_n), stay (p_{stay}), move south (p_s) and move east (p_e).

Some probabilities relevant to the BFTM are discussed below. [8]

1) The probability of distractor at position k, is P(d_k) = β.
2) The probability of firing of the k^{th} sensory neuron(S_k), either due to a fruit fly(f_k) or a distractor(d_k) is P(S_k | f_k, d_k) = α. The probability 'α' models the non-ideality of the sensory neuron.
3) If there is a fly in the receptive field of the k^{th} sensory neuron, then it will fire with a probability α, independent of what is in its receptive field.

$$L_{1k} = \begin{cases} \alpha, & S_k \text{ when } f_k \\ 1-\alpha, & \bar{S_k} \text{ when } f_k \end{cases}$$

4) If there is no fly in the k^{th} sensory neuron's receptive field, then there is a distractor with probability β. The marginal likelihood for no fly in this state are:

$$L_{0k} = \begin{cases} \alpha\beta, & S_k \text{ when } \bar{f_k} \text{ and } d_k \\ 1-\alpha\beta, & \bar{S_k} \text{ when } \bar{f_k} \text{ and } d_k \end{cases}$$

B. Bayesian Inference

The aim of the BFTM is to estimate the probability that the fruit fly is at position *k* given the sensory neuron at position *k* fires, i.e., $P(f_k|S_k)$.

$$P(f_k|S_k) = \frac{P(S_k|f_k).P(f_k)}{P(S_k)} = \frac{L_{1k}.P(f_k)}{L_{1k}.P(f_k) + L_{0k}.P(\bar{f_k})} \quad (1)$$

Here, $P(f_k|S_k)$ represents the required posterior probability, $P(f_k)$ represents the prior probability that the fly is present at position *k*, L_{0k} and L_{1k} represent the likelihood functions discussed in Section II-A.

C. Proposed System Overview

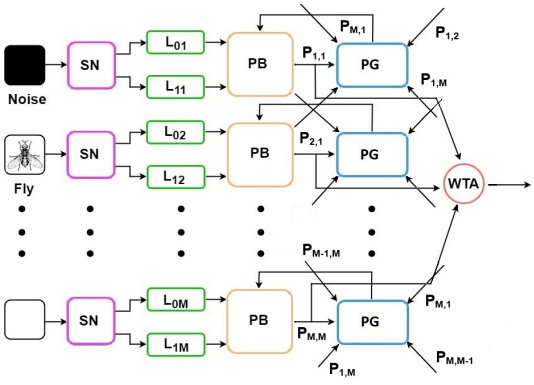

Fig. 2: **BFTM architecture:** SN, PB, PG and WTA are Sensory neuron, Posterior Block, Prior Generation Block and Winner Take All Block respectively. L_{0k} and L_{1k} represents the Likelihood Generator Block. Here, the probabilities $P_{j,k}$ represent the posterior probability of the $(j,k)^{th}$ path.

Proposed architecture for the Bayesian Fly Tracker is shown in Figure 2. There are *M*x*M* fly positions (the dragonfly's field of view is divided into *M*x*M* regions) and *M*x*M* parallel *inference paths*, one corresponding to each fly position. Each *inference path* consists of a *Sensory Neuron (SN)*, a pair of *Likelihood Blocks*, a *Posterior Block* and a *Prior Generation Block*. *Sensory Neurons* produce a spike when a fly or a distractor is present at the sensor position. This spike is fed to a pair of *Likelihood Blocks*, each of which computes the likelihoods L_{0k} and L_{1k}. Output of the *Prior Generation Block* is the prior probability of each state. These prior probabilities are fed back to the *Posterior Block* whose other input comes from the *Likelihood Block*. *Posterior Block* uses these inputs to compute the posterior probability of each state using Bayes' theorem. The $(j,k)^{th}$ *Prior Generation Block* takes the posterior probability computed in the $(j,k)^{th}$ position and in the four adjacent positions namely (*j-1,k*), (*j,k-1*), (*j+1,k*) and (*j,k+1*) as inputs to compute the prior probability for the next time step. Posterior probability computed in each path is fed to a *Winner Take All Block* which estimates the fly position using the Maximum A Posteriori (MAP) probability.

D. MATLAB Simulation Benchmark

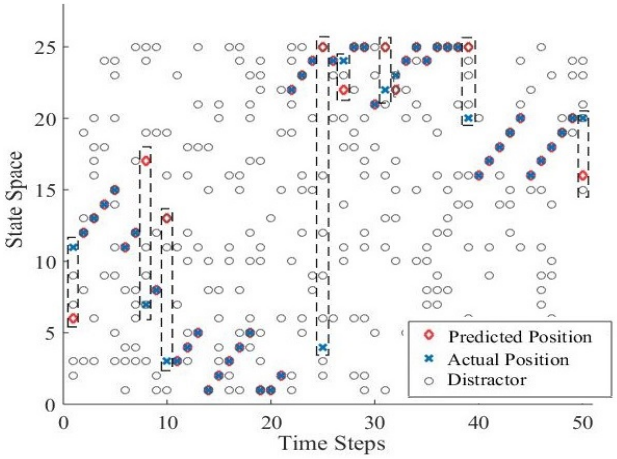

Fig. 3: **BFTM Simulation Result:** The mismatches between the actual position ('x') and the predicted positions ('◊') are highlighted using a rectangular box. In this figure, the two dimensional 5x5 grid is linearized as a state space, **S = 5*(X-1) + Y**.

	$\alpha = 0.8$	$\alpha = 0.85$	$\alpha = 0.9$	$\alpha = 0.95$
$\beta = 0$	96	98	98	98
$\beta = 0.1$	70	82	88	90
$\beta = 0.2$	66	70	74	82

TABLE I: Accuracy (%) as a function of α for different values of β

To study the performance of the BFTM, the model was implemented in MATLAB. The BFTM achieves an accuracy of 82% (refer Figure 3). The simulation was carried out with α=0.95 and β=0.2. The movement of the fruit fly is completely random subject to the probabilities $\{p_w, p_n, p_{stay}, p_s, p_e\}$. The transition probabilities $\{p_w, p_n, p_{stay}, p_s, p_e\}$ were chosen to be {0.05, 0.05, 0.15, 0.05, 0.7} for the inference, but can be learnt using maximum likelihood estimates. In this work, only the implementation of the inference path is shown.

III. Circuit Implementation

In the circuit implementation, the probabilities are represented by currents and a normalization current I_{norm} corresponds to a probability of one. From equation (1), it can be

seen that the circuit must perform mathematical operations like addition, multiplication and division. Addition of two currents is achieved by simply shorting the two branches. The other building blocks (described briefly in Section II-C) are discussed below.

A. Analog Multiplier/Divider Circuit

Figure 4 shows the circuit diagram of an analog multiplier/divider operating in weak inversion region. Transistors *M0-M3* form a closed loop in which *M0* and *M1* are connected in counter-clockwise direction and *M2* and *M3* are connected in clockwise direction. Applying trans-linear principle [11] to this loop,

$$I_{out} = \frac{I_1 I_2}{I_3} \tag{2}$$

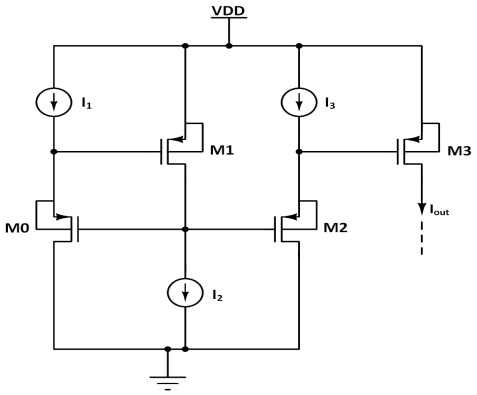

Fig. 4: Analog current multiplier/divider (CM/CD) circuit; reproduced from [12]

B. Likelihood Block

As defined in Section II-A, the likelihood L_{1k} is P($S_k|f_k$) when there is a spike and is $P(\overline{S_k}|f_k)$ when there is no spike. So, the behaviour of the likelihood block is similar to a current multiplexer which selects one out of the two currents at a time and the spike input acts as the select line. In Figure 5, transistors *M0-M3* acts as switches and transistors *M4-M7* and *M8-M11* form cascode current mirrors. I_{in1} flows to the output when spike input is high and I_{in2} otherwise.

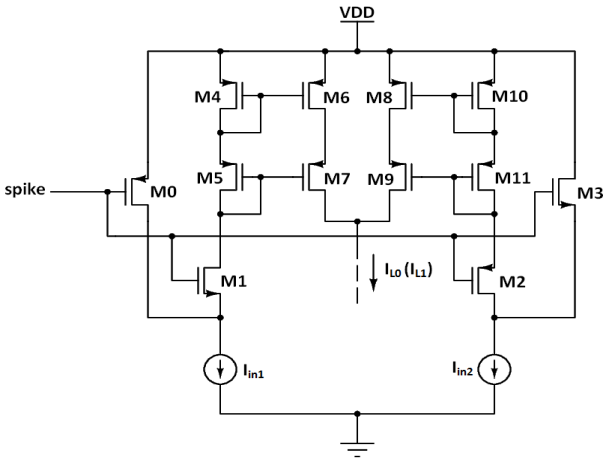

Fig. 5: Likelihood Block Circuit Implementation

C. Posterior Block

Posterior Block calculates the posterior probability corresponding to each fly location from the prior probabilities and likelihood function using Baye's equation. So, the *Posterior Block* is basically the implementation of Baye's equation, which is given below.

$$I_{post} = \frac{L_{1k}.P(f_k)}{L_{1k}.P(f_k) + L_{0k}.P(\bar{f}_k)} \tag{3}$$

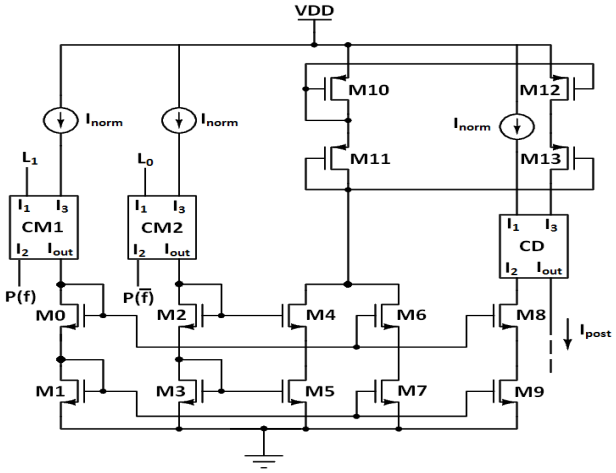

Fig. 6: Posterior Block Circuit Implementation

Figure 6 shows the circuit implementation of equation (3). Here, *CM1*, *CM2* and *CD* represent the analog multiplier/divider circuit presented in Section III-A. As shown in the figure, L_1 and P(f) are the input currents to *CM1*. L_0 and P($\overline{f}$) are the input currents to *CM2*. Outputs of *CM1* and *CM2* are added to form the denominator whereas the output of *CM1* itself forms the numerator. *CD* performs the division operation. I_{norm} is applied to the third input of multiplier and divider for the purpose of normalization.

D. Prior Generation Block

At the outset, all the *M*x*M* paths have the same prior probabilities. But, as the fruit fly moves, the prior probability of each path gets modified. The purpose of the *Prior Generation Block* is to re-calculate the prior probabilities for each of these paths. Since the fruit fly's movement is limited to 2-D as discussed in Section II-A, the prior probability is given by

$$\begin{aligned} p_{prior}(x,y) = {} & p_w * p_{post}(x,y+1) + p_n * p_{post}(x+1,y) \\ & + p_{stay} * p_{post}(x,y) + p_s \\ & * p_{post}(x-1,y) + p_e * p_{post}(x,y-1) \end{aligned}$$

where $p_{prior}(x,y)$ denotes the generated prior, $p_{post}(x,y)$ denotes the posterior probability of the (x,y) path and $\{p_w, p_n, p_{stay}, p_s, p_e\}$ denotes the transition probabilities discussed in Section II-A.

Figure 7 shows the circuit implementation of the *Prior Generation Block*. In the figure, *CM0* - *CM4* represent the current multipliers used to carry out the multiplications in equation (III-D) and *TX0* - *TX2* represent transmission gates. The aim is to feedback this current (I_{prior}) to the *Posterior*

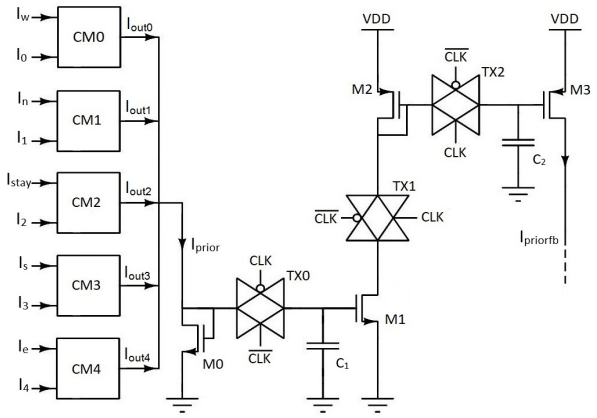

Fig. 7: Prior Generation Block Circuit Implementation

Block, with a delay of one clock cycle. This delay is necessary for the stability of the feedback loop (*Prior Generation Block* to *Posterior Block*).

When the *CLK* is low, *TX0* is ON and the gate voltage that causes I_{prior} to flow through *M0* is sampled onto capacitor *C1*. During this interval, both *TX1* and *TX2* are OFF and no current flows through *M1-M2*. When the *CLK* goes high, *TX0* is OFF and both *TX1* and *TX2* turn ON. During this interval, I_{d1} = I_{d2} = I_{prior} provided *M0* and *M1* are matched ($(W/L)_0$ = $(W/L)_1$). Also, the gate voltage that causes I_{prior} to flow through *M2* is sampled onto capacitor *C2* and I_{d3} = I_{d2} = I_{prior} = $I_{priorfb}$, provided *M2* and *M3* are matched ($(W/L)_2$ = $(W/L)_3$). The capacitor *C2* ensures that the output current, $I_{priorfb}(k)$ = $I_{prior}(k-1)$ over the entire clock.

E. Winner Take All Block

The *Winner Take All(WTA) Block*, as the name suggests, determines which of the *M*x*M* paths have the maximum posterior probability and thereby determines the most likely position of the fruit fly at a given point in time. The circuit is reproduced from [13], [14]. The operation of the *Winner Take All Block* may be summarized as,

$$I_{outX} \approx I_b, \text{ if } I_{inX} > I_{ink}\ \forall\ \text{k} \neq \text{X}$$

IV. Post Layout Simulation Results

The complete system was implemented in TSMC 180nm technology and the tracking performance was verified by carrying out the post layout simulation in Cadence Virtuoso environment. A constrained spike input (subject to probabilities α, β, p_n, p_w, p_{stay}, p_e and p_s) is fed as the input to both the software tracking model (implemented in MATLAB) as well as the circuit implementation. To quantify the accuracy, the outputs of the circuit implementation (post-layout) and the MATLAB model are compared. Figure 8 shows the layout of the entire design. The total area of the design is $0.067mm^2$. The maximum clock frequency of the system is 250 KHz. If the clock frequency exceeds 250 KHz, then the output current of the *Prior Generation Block* fails to settle, thereby reducing the accuracy.

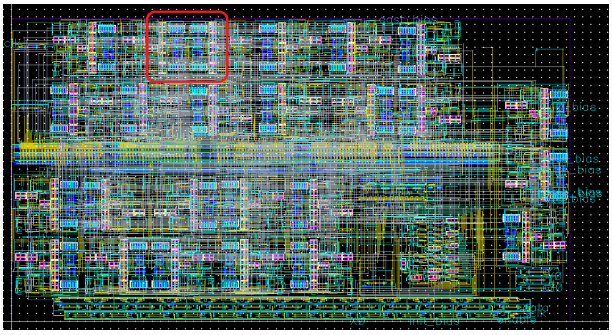

Fig. 8: Layout of the entire design. The red box shows the layout of a single inference path.

A. Trade-off between accuracy and power

Table II shows the accuracy as a function of the normalization current. The accuracy increases with normalization current for $I_{norm} \leq$ 200nA and decreases for $I_{norm} >$ 200nA. For small values of I_{norm}, the error due to the offset introduced in the fedback current (by the *Prior Generation Block*) constitutes a larger percentage thereby reducing the accuracy. As I_{norm} is increased, the multiplier becomes non-linear thereby reducing the accuracy. Table III shows the accuracy for different values of transition probabilities.

I_{norm} (nA)	50	75	100	150	**200**	250
Circuit Accuracy (%)	70	72	76	80	**82**	76

TABLE II: Accuracy as a function of the normalization current

p_n	p_w	p_{stay}	p_e	p_s	Matlab (%)	Circuit (%)
0.1	0.35	0.3	0.05	0.2	92	90
0.05	0.05	0.15	0.7	0.05	82	82
0.3	0.1	0.05	0.4	0.15	78	78

TABLE III: Accuracy (%) for different transition probabilities using I_{norm} = 200nA

V. Conclusion

The present work shows the implementation of a Bayesian Fly Tracking Model using analog neuromorphic circuits. Since all the transistors used for computation are operating in weak inversion region, currents are in the range of few hundreds of nA ($\leq$200nA for maximum accuracy). Maximum accuracy of 82% (same as that of the software model) was achieved for a power of 0.15mW (for a 5x5 grid with clock frequency = 25 KHz). As discussed in Section IV-A, accuracy depends on the normalization current I_{norm}. As I_{norm} is increased from 50nA to 200nA, accuracy increases from 72% to 82%. This presents a trade-off between power and accuracy and the resulting choice of normalization current (I_{norm}) depends on the application at hand. The design of a 5x5 grid occupies an area of $0.067mm^2$ which is considerably small. The existing system can be scaled from *M*x*M* paths to *P*x*Q* paths with very little effort, as scaling up requires only replication of the existing fundamental blocks. The proposed system would help to develop more complex probabilistic neuromorphic algorithms on analog VLSI substrate for low power applications.

REFERENCES

[1] Paulin M. G., "Bayesian tracking of a moving target in a cluttered environment using spiking neurons", *NeuroEng*, 2014.

[2] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, P.J. Nordlund, "Particle filters for positioning, navigation and tracking", *IEEE Transactions on Signal Processing*, 2002.

[3] Kang Li, Fazhi He, Haiping Yu, Xiao Chen, "A parallel and robust object tracking approach synthesizing adaptive Bayesian learning and improved incremental subspace learning", *Frontiers of Computer Science*, 2019.

[4] Jennifer L. Palmer, Ricardo Cannizzaro, Branko Ristic, "Source Localisation with a Bernoulli Particle-Filter Based Bearings-Only Tracking Algorithm", *Defence Science and Technology Group, Australia.*

[5] Alessio Dore, Mauricio Soto, Carlo S. Regazzoni, "Bayesian Tracking for Video Analytics", *IEEE Signal Processing Magazine*, 2010.

[6] Chakrapani L. N., Korkmaz P. and palem K. V. ,"Probabilistic system on a chip architectures", *ACM Transactions on System Automation and Electronic Systems*, 2007.

[7] Murray A. F. ,"Pulse arithmetic in VLSI neural networks", *IEEE Micro*, 1989.

[8] Chetan Singh Thakur, Saeed Afshar, Runchun M. Wang, Tara J. Hamilton, Jonathan Tapson and Andre van Schaik, "Bayesian Estimation and Inference using Stochastic Electronics", *Frontiers in Neuroscience*, 2016.

[9] Thakur, C. S., L. Molin, A. V. Schaik, R. Etienne-Cummings, "Inference in Spiking Bayesian Neurons using Stochastic Computation", *Information Sciences and Systems (CISS), 51st Annual Conference on, IEEE*, 2017.

[10] Paulin M. G. and van Schaik ,"Bayesian Inference with spiking neurons", *NeuroEng*, 2014.

[11] Remco J. Wiegerink, "The MOS Translinear Principle", *Analysis and Synthesis of MOS Translinear Circuits, Springer Science*, 1993.

[12] Liu, Kramer, Indiveri, Delbruck and Douglas, "Analysis and Synthesis of MOS Translinear Circuits", *Analog VLSI : Circuits and Principles, MIT Press*.

[13] Lazzaro J., Ryckebusch S., Mahowald M. A., Mead C. A., "Winner-take-all networks of O(n) complexity, Advances in Neural Information Processing Systems", *Morgan Kaufmann*, 1989.

[14] Choi J., Sheu B. J., "A high precision VLSI winner-take-all circuit for self-organizing neural networks", *IEEE Journal of Solid-State Circuits*, 28(5):576–584, 1993.