BIO-INSPIRED OBSTACLE DETECTION SYSTEM

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea "
Politehnica" din Timişoara
în domeniul INGINERIE ELECTRONICĂ ŞI TELECOMUNICAŢII
de către

Ing. Mihai Emanuel Basch

Conducător ştiinţific:	prof.univ.dr.ing. Virgil Tiponuţ
Referenţi ştiinţifici:	prof.univ.dr.ing. Liviu Goraş prof.univ.dr.ing. Gavril Toderean prof.univ.dr.ing. Corneliu Toma

Ziua susţinerii tezei: 28.09.2012

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Ingineria Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie Mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor
12. Ingineria sistemelor
13. Inginerie energetică
14. Calculatoare şi tehnologia informaţiei

Universitatea „Politehnica" din Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300159 Timişoara, Bd. Republicii 9,
tel. 0256 403823, fax. 0256 403221
e-mail: editura@edipol.upt.ro

PREFACE

This thesis presents a concept of a bio-inspired obstacle detection system, mainly based on insect vision that can be used as a guidance device for people with visual disabilities or in automotive industry as an additional safety measurement. The proposed system's functionality is evaluated through theoretical analysis and simulation with real life situations. There is also a proposal for a hardware type implementation of the system into a single CMOS chip and a software based implementation using a smartphone device.

Outline of the thesis

The thesis is organized in five chapters as follows:

The *first chapter* presents a biological background for the insect based vision, in particular is described the fly and the locust visual system since these two insects represents the main inspiration for the bio-inspired obstacle detection system.

The *second chapter* contains an overview on existing bio-inspired obstacle detection systems that tend to be a "state of the art" on this subject that is not very researched in the literature since this is a niche domain.

In *the third chapter* are analyzed a few new concepts for a bio-inspired motion and obstacle detection system and evaluated through simulation using real life situations.

The *fourth chapter* contains two solutions for the implementation of the bio-inspired obstacle detection: a hardware type implementation into CMOS chip and an embedded type using a smart-phone device.

In *the fifth chapter* are presented the conclusions regarding the applicability of the bio-inspired obstacle detection system and also the original contribution of this work are summarized.

Acknowledgement

This work was partially supported by the strategic grant POSDRU/88/1.5/S/50783, Project ID50783 (2009), co-financed by the European Social Fund – Investing in People, within the Sectoral Operational Programme Human Resources Development 2007-2013.

Timisoara, September 28th , 2012

Mihai Emanuel Basch

Mihai Emanuel, Basch

INSECT BASED VISION BIO-INSPIRED OBSTACLE DETECTION SYSTEM

Teze de doctorat ale UPT, Seria 7, Nr.47, Editura Politehnica, 2012, 122 pagini, 72 figuri.

ISSN: 1842-7014

ISBN: 978-606-554-512-0

Keywords: Insect, Vision, EMD, Obstacle, Detection, Bio-inspired, LGMD, Chip, VLSI, Android

Objectives of the thesis

The main objectives of the thesis are:

- To present an overview on existing bio-inspired obstacle detection systems;
- To present an enhanced system of a bio-inspired motion and obstacle detection system that can be used as an assitive device ;
- To evaluate the functionality of the proposed bio-inspired motion and obstacle detection system;
- To propose solutions for the implementation of the bio-inspired obstacle detection system

Table of contents

Abbreviations

CMEMD – Current Mode Elementary Motion Detector;
CMLPF – Current Mode Low Pass Filter;
CMOS – Complementary Metal Oxide Semiconductor;
DCMD – Descending Contra-Lateral Movement Detector;
EMD – Elementary Motion Detector;
FIS – Fuzzy Inference System;
GND – Ground connection;
GUI – Graphical User Interface;
HS – Horizontal System;
HW – Hardware;
LGMD – Lobula Giant Motion Detector;
LPF – Low Pass Filter;
NMOS – N-channel Metal Oxide Semiconductor;
PMOS – P-channel Metal Oxide Semiconductor;
SW – Software;
VLSI – Very Large Scale Integration;
VMEMD – Voltage Mode Elementary Motion Detector;
VS – Vertical System.

List of figures

1 Insect based vision

In order to understand how to design an artificial system that emulates biological functions it is necessary to know a few details about it. This chapter presents an overview about the insect visual functions especially for some species of flies and locust which are the main inspiration for this thesis in order to implement a bio-inspired obstacle detection system.

1.1 Introduction

During centuries, insects have extended into thousands of types, accounting for more than half of all known residing organisms: they are the very best group in the animal world, discovered in almost all conditions around the globe, with a body size ranging from as small as 1 mm into as large as about a half meter [17]. Their visual organs, together with the specific parts of the neurological system devoted to the handling of visible information, have long been the topic of extreme research but, excluding some very primary reactions, it is still not possible to link an insect's visible feedback to its behavior outcome [17]. The most intense research was made on Drosophila fruit fly. This biological machine maintains great guarantee for the insect perspective field, offering the possibility of controlling or exciting any single neuron in its neurological system. Insects are considered to exist even before dinosaurs. The visual structure of insects have attracted researchers and physicists for decades, because insect activities is sometimes so stereotyped as to suggest that the techniques handling this small representative can quickly be deduced. Nowadays, there is set attention in insect viewpoint, triggered by the ever-more modern toolset offered by Drosophila genetics [17].

1.2 Insect visual system

1.2.1 Insects eyes

The biological structure of insect eyes is different from the eyes of humans. They are designed from multiple elements called "facets". The eye types can be categorized as: the apposition eye (Fig.1.1.a), where the photoreceptors are optically isolated from each other; the superposition eye (Fig.1.1.b), where the facets are separated from the array of photoreceptors by a zone with many facets acting together as a single optical device [17], [18]. The superposition eye is typically found in night insects, while the apposition eye is found to daylight insects such as locusts, honeybees and flies. In the fly eye, six photoreceptors encompass two main ones that are placed one on top of the other [17], [18], [19]. These groups of photoreceptors are connected to the same postsynaptic element, and by this it is increased the sensitivity and the acuity also remain very high [20], [21]. This principle is called "neural superposition" (Fig.1.1.c), and it is found in dipteran flies. Even though image development appear to be very different in insect in comparison with vertebrates, in both circumstances the image is approximated onto the assistance of a variety of photoreceptors, forming a perspective of the environment. It has been many years misinterpretation that insects see the world in a pixilated form but despite of the discrete image representation by the photoreceptor array, the brain receive a single image [17], [21].

As a general observation for species with apposition eyes, if the animal is bigger, then it has larger eyes with the more facets and also a better spatial resolution. This fact is because if the lenses are smaller they will capture less light, and if in these conditions the number of facets were kept constant on a smaller surface would generate an image with additional noise and also, lenses to be functional, they need to have a minimum diameter.

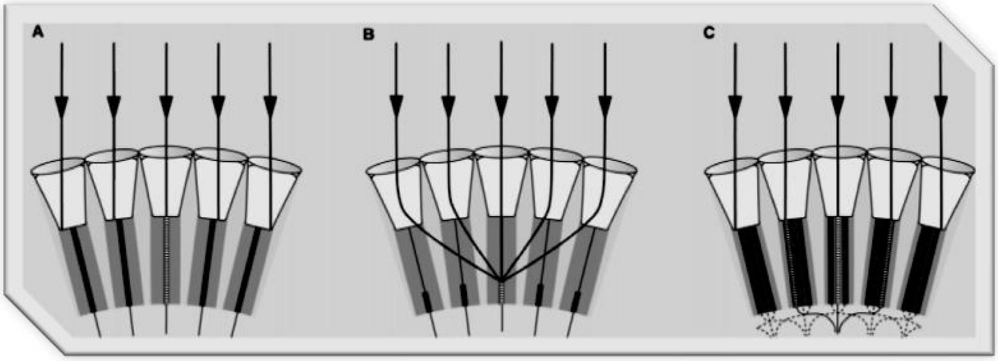


Fig. 1.1. Classification of insect eyes.

(A) Apposition eye. (B) Superposition eye. (C) Neural superposition eye. (Reproduced after after [17].)

The nervous system of insects consists of a head ganglion, three thoracic ganglia and a few abdominal ganglia. In Drosophila, the three thoracic and abdominal ganglia are mixed into one thoracic ganglion, which is related to the head ganglia by the cervical connective [17], [22]. The brain is composed from central brain, the sub-esophageal ganglion and the primary sensory centers such as the antennal lobes and the visual ganglia [17], [23]. The neurons are contained in a cortex encircling the ganglia, ramifying amongst each other. The visual ganglia is formed by the following biological layers: lamina, medulla and lobula complex. In particular cases such as the dipteran flies species, the lobula complex is also diveded into an anterior 'lobula' and a posterior lobula plate.

In Drosophila, the central photoreceptor cells are going through the lamina without making synapses and terminate in the medulla layer. Many types of neurons found in one species of insects are finded also in other types of insects and sometimes they are indistinguishable. In the lamina layer eight different cell types are found: the lamina monopolar cells, two centrifugal cells, and the so called T cells. All these cells connect the lamina with the medulla. Intracellular recordings from the monopolar showed a high-pass filtering of the signals in the lamina monopolar cells provided by photoreceptors [17], [24], [25]. The cells from the lamina are ramified in different layers from medulla [17], [26] showing that the photoreceptor signals split into parallel pathways. Between these types of cells, the intrinsic medulla or ‘Mi’ neurons ramify into discrete layers of the medulla. The trans-medulla cells are a type of columnar neurons and their roles is to connect several distinct layers of the medulla to the lobula and are considered to be motion-sensitive. They are grouped into different classes, such as the horizontal system (HS-cells) and the vertical system (VS-cells).

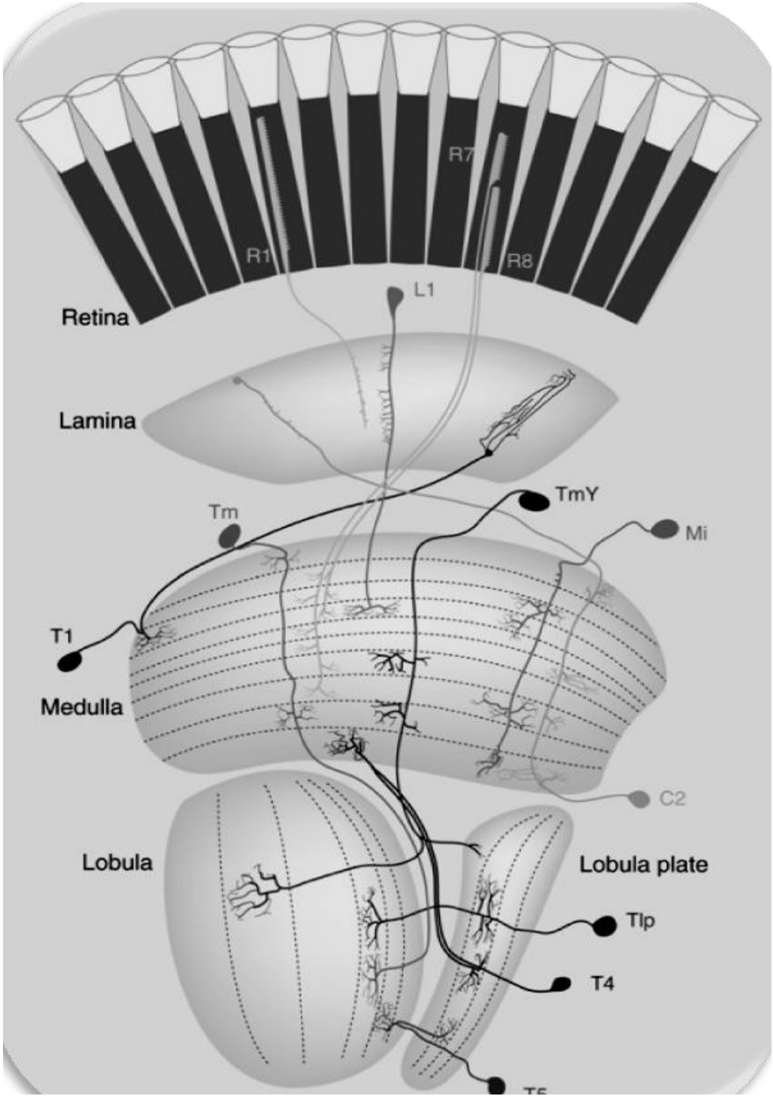


Fig. 1.2. Visual system of the fly. Reproduced after [17].

The insect based vision is diversified according with their eye morphologies. Based on which parameter of the light is being used or what details is produced from the main neurological details, perspective is often separated into subcategories like polarization perspective, shade perspective, level understanding and movement perspective. For example, based on the position of the sun as a reference the honeybees inform their group members about the location of a food source by their waggle dance [17], [27], [28]. The house-flies expose methodical tendencies in response to slow rotation of the plane of polarisation without any intensity gradients present in the stimulus [17], [28]. Another example is the dung beetles (Scarabeus zambesianus) which uses skylight polarisation to maintain and stabilize a straight course [29]. All these behaviors are related with the detection of the polarisation plane by an array of photoreceptors which are found in the facet eye along the dorsal rim.

Physical objects separate from the background by relative motion in both cases, when they themselves are in motion, or when the viewer is moving, revealing in this way the three-dimensional structure of the world. Experiments on different type of species revealed these particularities. For example the wasps (Odynerus spinipes), performs characteristic zig-zag flights when they are in front of an object in order to [17], [31] to stabilize the relative motion between object and background, another example is the locust (Schistocerca americana) who use motion parallax in order to aim the jump trajectory to a far away target platform [32]. Tiger beetles [17], [33], [34] are known to follow their prey visually, and also other various fly species.

Object approach generates an optic flow with a strong expansion in front of the animal and is a sign for a possible collision [17], [35]. For example, an approaching object generates an escape jump in locusts [17], [36], [37]. A optic flow that is rotating around the vertical body axis may indicate an involuntary rotation opposite to the flow, leading to compensatory steering maneuvers of the animal direction with the motion of the surround. This reaction, the opto-motor response is found in almost every animal investigated so far, from insects to fish to man, and has been seminal in the discovery of the elementary mechanism of motion detection (for review see [38]).

Starting with experiments made on beetles (Chlorophanus viridis) and house-flies, Reichardt and his team [17], [39], [40] developed a computational model of an elementary motion detection in which the signal processing from the luminance input encodes the motion objects from the visual field. If one sums up the output signals of a two-dimensional array of such 'Reichardt detectors', the resulting value describes the steady-state optomotor response as a function of several stimulus parameters in surprising detail [17], [41]. Because of the small size of most of the columnar neurons in the optic lobes, however, it is still an open question which cells represent the various processing stages in the Reichardt detector.

1.2.2 Particularities of the locust visual system

Many creatures start to avoid by going away a risk from the moment it is recognized. It has been highlighted in some experiments that the escape jumps of locusts are producing with a few hundred of milliseconds before an obstacle appears. Directly responsible for this reaction is supposed to be mainly two important neurons that were found in locust: DCMD (descending contra-lateral movement detector) and LGMD (lobula giant motion detector). A sudden movement of small objects in the visual field can trigger the locust to move away. According to (Burrows and Rowell, 1973) the DCMD and also its homologue, the DIMD, with meta-thoracic moto-neurons are involved in the jumping process [42].

Spikes that appear in the LGMD are followed by those in the DCMD at frequencies up to 400 Hz [37] - [42], [47]. It has been confirmed that the response of these neurons are much sensitive for the approaching objects in comparison with the translating objects [42], [48]."The DCMD spiking response declines to half the maximum level if the object’s trajectory deviates from a direct collision course with tuning slightly tighter in the azimuth than in the elevation” (Fig. 2C).

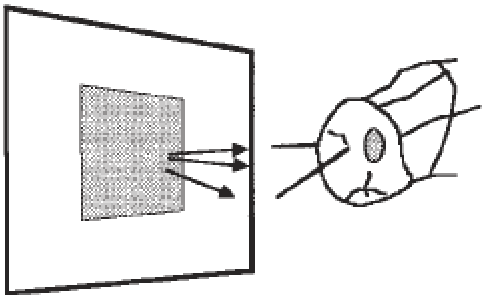


Fig. 1.3. The DCMD and LGMD neurons are tuned to detect objects on the collision course. Reproduced from [43].

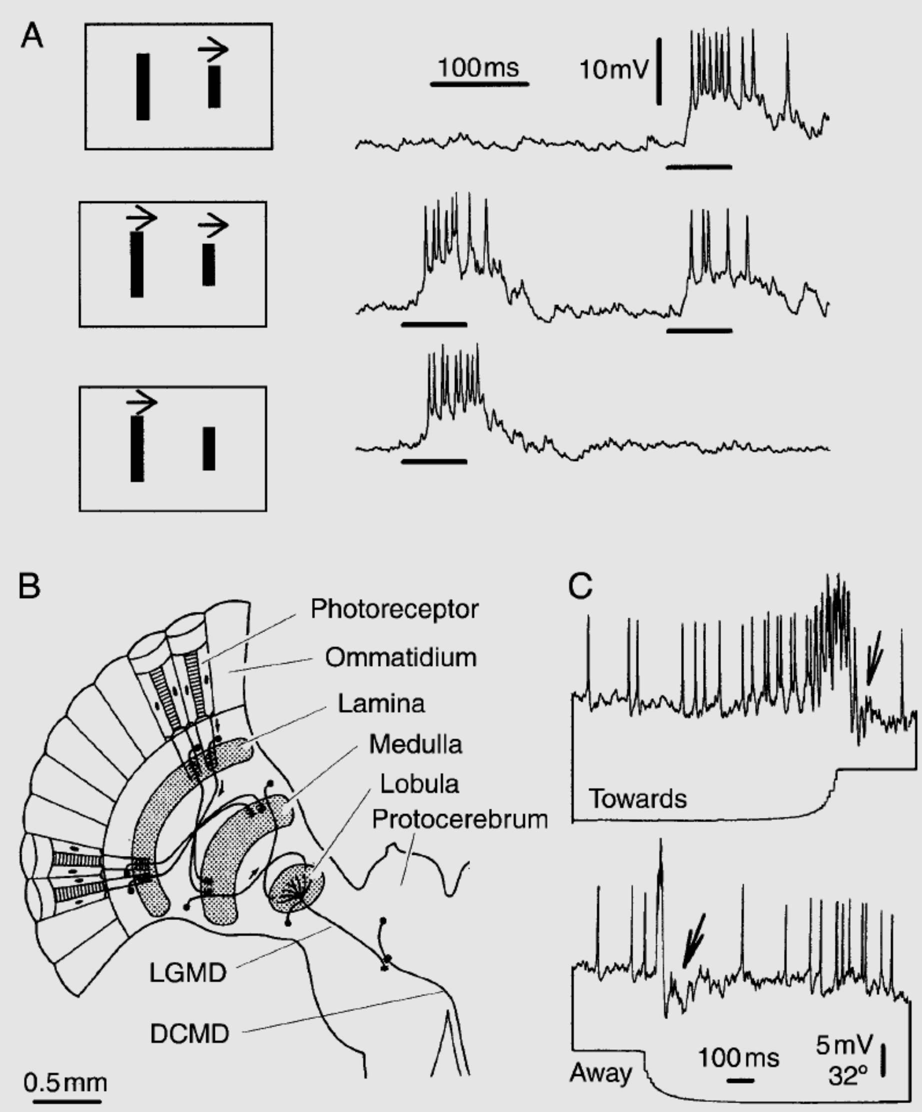
The response of the LGMD during the approach of an object depends on both the size and the speed of it, this is way it is considered that it acts in pair with other neurons because only alone it could not signal the remaining time to collision [42], [43]. Even if in theory it was proposed that the time to collision could be approximated by an exponential function:

$$f(\theta) = C \cdot \theta' \cdot e^{-\alpha\theta}$$

where “θ” is the angular extent of the image, θ’ is the instantaneous rate of change of “θ”, and “C” and “α” are constants.

The issue regarding this function is that in practical experiments was not achieved. Other experiments like the ones with lines that changed in length and also in speed in front of the locust eyes showed an increased sensitivity and because of that it was concluded that the LGMD neuron is stimulated by the number of edges of an object either if the edges are converging or diverging [52]. “Changes in luminance, caused by object motion, generate little or no response and the neurons detect the approach of light or of dark objects” [42].

The preferences of the LGMD for objects that are approaching rather than are moving away is considered to be a result of a critical competition over the dendrites of the LGMD in the optic lobe [43], [44], [51]. This competition is between the excitation, determined by the edges of the expanding image in motion, and the lateral inhibition that is mediated by synapses onto neurons found in the medulla. Powerful responses from the DCMD and LGMD appear when excitation comes before inhibition and wins this competition [42] - [49], [50].



a) b)

Fig. 1.4. Experiment with LGMD neuron -Intracellular recordings that show two forms of inhibition.

(A) The locust view two vertical bars moved laterally on a screen. Movement of the right bar alone caused vigorous excitation of the LGMD (top trace), which was reduced if the left bar moved before the right one (middle trace). Following movement of the left bar alone, no IPSPs were recorded in the LGMD (lower trace), indicating that the inhibitory effect occurred presynaptically to the LGMD. (B) Schematic illustration of the locust visual system viewed from behind. (C) The locust viewed the image of a rectangular object approaching and later retreating at 5 m/s. Following the end of approach and the start of retreat, which are times when large parts of the visual field are stimulated rapidly, a barrage of IPSPs was recorded (arrows). Reproduced from [43]

2 Overview on existing bio-inspired obstacle detection systems

This chapter presents an overview on the bio-inspired motion and obstacle detection systems available on the literature. The most studied insects in this sense are the fly and locust, many experiments were made and some projects that follow their principle were made and will be described next.

2.1 Bio-inspired motion detection based on insect vision

Even if insects are considered to be much simpler organisms in comparison to vertebrates and other species, having a nervous system with a smaller size and less complexity, they perform a very good job of flying and other tasks such as detection and tracking objects, all of these based on a low-resolution visual sense. That is way insects can be considered to be a great source of inspiration for artificial system that mimic these types of functions [53], [54], [55].

The visual system of the insects has a hierarchical organization in which every biological layer is supposed to has an own function. The insects (in particular the Fly and the Locust which are a source of inspiration in this case) have a multi-lens construction of the compound eye, but the image projected onto the underlying retina is a single image of the visual field. The photoreceptors contained into the retina adapt to the ambient light level, and any deviation from this level is further signaled to the next layer of cells, called the lamina. "Lamina cells generally show transient or high pass responses, emphasizing temporal change" [82].The next step in the visual processing is happening in the medulla. Because the cells in the medulla are very small, it is quite difficult to study their exact function. Different experiments showed that local measures of motion (i.e., between adjacent photoreceptors) are computed here. These measures of motion are integrated after that by tangential cells located in the lobular plate [82], [102], [104]. For example, the house fly it is supposed to have about 50-60 tangential cells in each half of its brain. "Lobular plate cells generally respond to motion over large parts of the visual field. Some of these cells seem to be matched filters for the optic patterns produced by rotation or translation along particular axes (Krapp and Hengstenberg, 1996)" [82].

The motion detection mechanism based on insect vision was put in evidence by Hassenstein and Reichardt and they called it: EMD (elementary motion detector). The architecture of the EMD is based on correlating the output of one photoreceptor with the delayed output of an adjacent photoreceptor [82], [103].

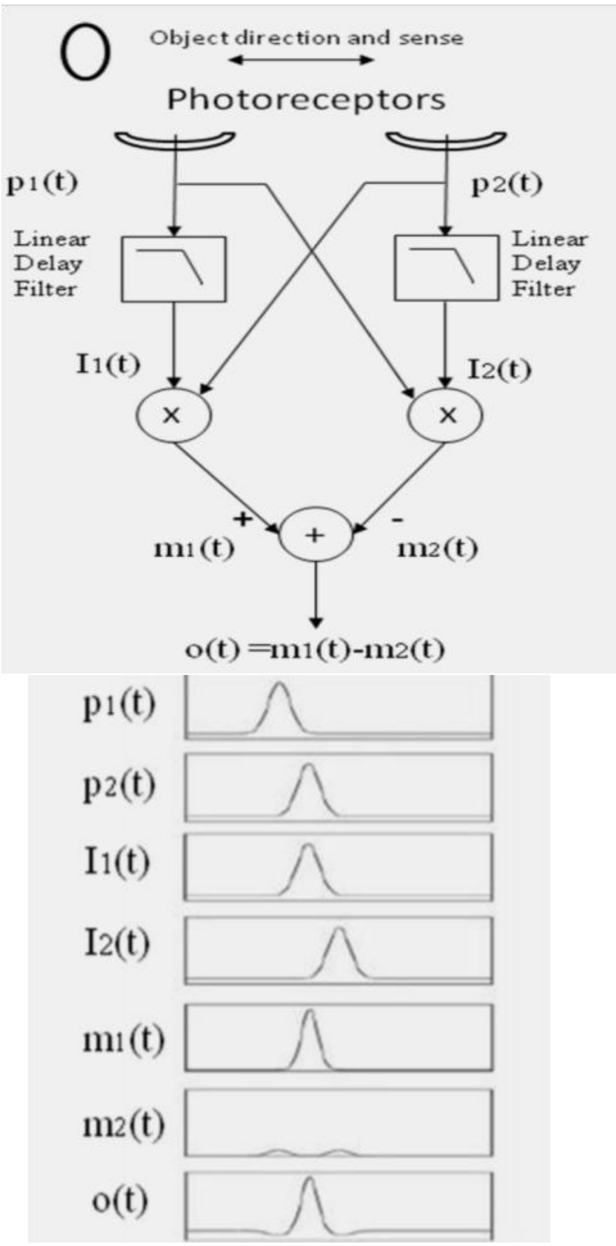


Fig. 2.1. Functional block diagram for the Elementary Motion Detector and the related signals for each block [7].

The EMD structure is showed in Fig.2.1 and is basically formed from two channels where the signal received from the photoreceptors is delayed in each channel and multiplied with the non-delayed signal from the opposite channel, then are subtracted in order to receive the motion information. If the result has a positive value means that an object has passed in front of the sensors from left to right and if the result has a negative value then the object has passed from right to left [7].

According to [7], if the input is a sinusoidal signal with wavelength λ that is moving with a speed v [º / s], then the image intensity I (x, t) can be written as:

$$i(x,t) = I + \Delta I \sin(2\pi f_s(x + vt)) \qquad (2.1)$$

where I is average intensity and spatial frequency fs. The contrast pattern is and ΔI/I. In each photoreceptor, this moving pattern produces a sinusoidal signal temporal with the frequency $f_t = vf_s$. Thus we can rewrite () as:

$$i(x,t) = I + \Delta I \sin(\omega_t t + \omega_s x) \qquad (2.2)$$

where $\omega_t = 2\pi f_t$ and $\omega_s = 2\pi f_s$. If two receivers have a separation angle of Φ, the signals measured by the two photoreceptors can be expressed as:

$$p_1 = |H(\omega_t)| \Delta I \sin(\omega_t t - \omega_s \Phi / 2) \qquad (2.3)$$

$$p_2 = |H(\omega_t)| \Delta I \sin(\omega_t t - \omega_s \Phi / 2) \qquad (2.4)$$

where we have introduced H(ωt) as the frequency response of the photoreceptors. If using low-pass filters of first order, we get signals:

$$I_1(t) = |H(\omega_t)| \Delta I \left(t^2 \omega_t^2 + 1\right)^{-0.5} \sin\left(\omega_t t - \omega_s \Phi / 2 - \tan^{-1} \tau\omega_t\right) \qquad (2.5)$$

$$I_2(t) = |H(\omega_t)| \Delta I \left(\tau^2 \omega_t^2 + 1\right)^{-0,5} \sin\left(\omega_t t - \omega_s \Phi / 2 - \tan^{-1} \tau\omega_t\right) \qquad (2.6)$$

Correlation is performed by multiplying the out of phase signals with adjacent non-delayed signals. The results are:

$$m_1(t) = G\left[\cos(\omega_s \Phi + P) - \cos(2\omega_t t - P)\right] \qquad (2.7)$$

$$m_2(t) = G\left[\cos(\omega_s \Phi + P) - \cos(2\omega_t t - P)\right] \qquad (2.8)$$

where G and P are:

$$G = \left(|H(\omega_t)| \Delta I\right)^2 \left(2\left(\tau^2 \omega^2 + 1\right)\right)^{-0,5} \qquad (2.9)$$

$$P = \tan^{-1}(\tau\omega_t) \qquad (2.10)$$

So, the final answer becomes:

$$o(t) = (\Delta I)^{2} \left|H(\omega_t)\right|^{2} \tau\omega_\tau \left(\tau^{2}\omega_t^{2} + 1\right)^{-0,5} \sin(\Phi\omega_s) \tag{2.11}$$

If an object is passing across the photoreceptors, it produces a strong response when it first passes over the delayed photoreceptor, then in front of the non delayed photoreceptor, so the maximum response is when the transit time between two photoreceptors is equal with time delay of the elementary motion detector [7].

2.2 Obstacle detection system inspired from the locust LGMD neuron

A collision detection system based on a neural network implementations inspired from the Lobula Giant Motion Detector (LGMD) found in locusts was proposed in [4]. LGMD is a large neuron found in the optical lobule of the locust that mainly responds at the approaching objects [2], [56], [57]. Using electro-physical techniques, C. Rind formulated the functional structure and a mathematical description of the LGMD neuron [58].

This neuron reacts on objects that are moving on direct collision course and gives few or no response for receding objects [2], [57]. The neuron responds with a train of pulses that increase in frequency when the object is approaching. Rind proposed such a model of a neural network with four hierarchical levels obtained after several experiments with locusts [58].

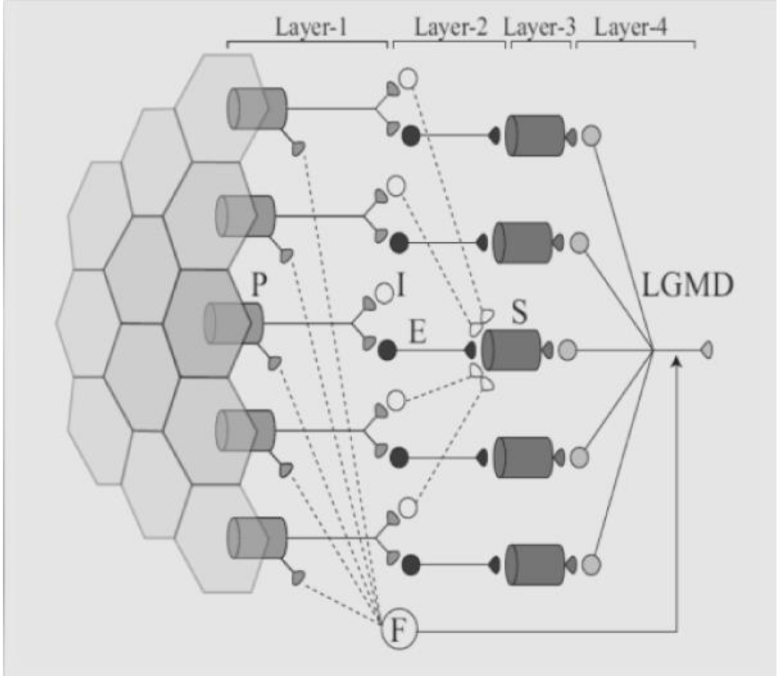


Fig. 2.2. The model of the LGMD neural network proposed by Rind [4].

In Fig.2.2 is presented the model with the specific layers. The first level represents the neuronal network input and is formed by a group of units called "P units" that react at changes of illumination in the scene as a result of the object edges being in motion. Using a high-pass temporal filter, the value of the illumination from the anterior moment is subtracted from the actual value of illumination. This creates an excitation signal that is transferred to the next hierarchical level composed by units called "E type" that send the excitation signals to the next level, while other units called "I type" units transmit delayed inhibitory signals to the closed neighbors. In the next level of units, "S type" units, the inhibitory signals are being subtracted from the excitatory signal corresponding of the same level. Each output of the "S type" units are summed in the LGMD neuron of the fourth level, and when this value exceeds a fixed threshold, the LGMD neuron generates a spike. Based on how many spikes are generated it can be find if an imminent collision can occur. The "F type" units composed by the summing of the "P units" have the role of creating a feed forward inhibition signal in order to prevent the LGMD neuron to respond to background illumination changes as a false collision. Based on the equations described in [2], [59], in Fig.2.3 is presented a simulated situation of the LGMD model developed in MATLAB according with [2]. In the considered situation, a person is approaching on direct collision course to the system in an environment with a simple background for this particular and only the person is moving, while the system is stationary.

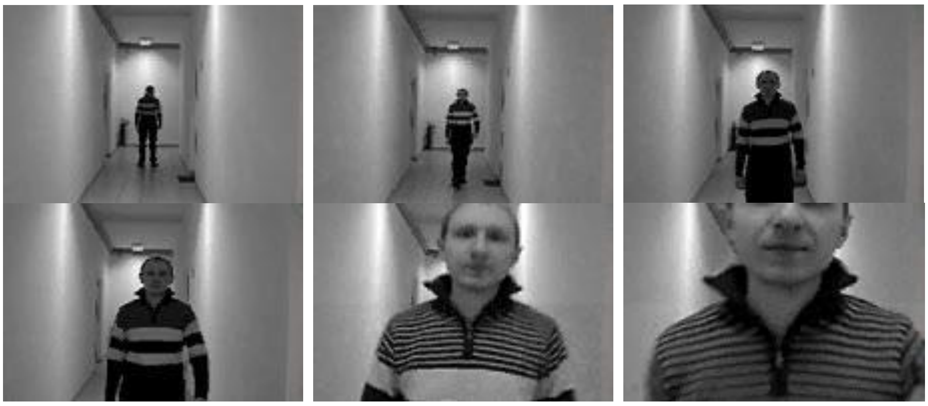


Fig. 2.3. Snapshots taken from the recording used as input for the Neural network [4]

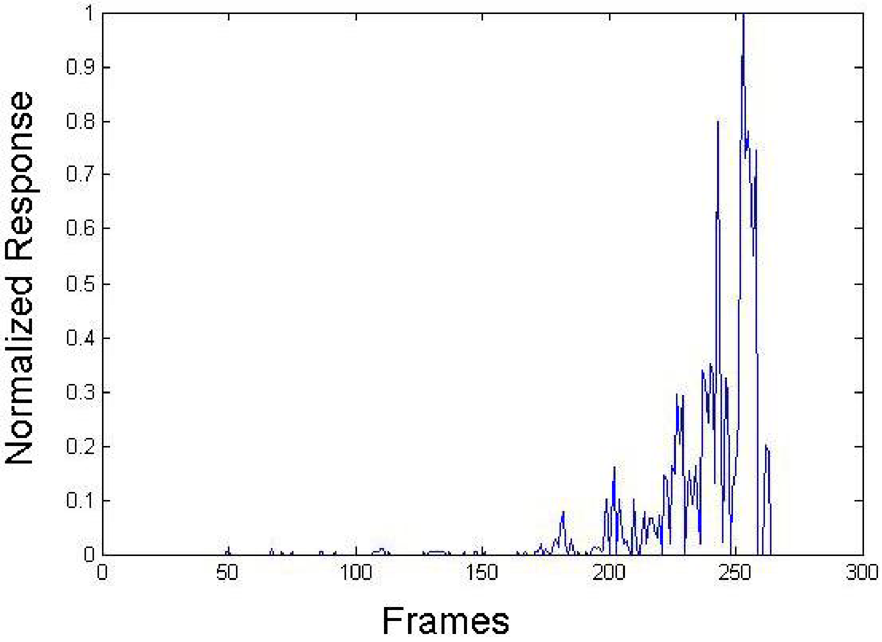


Fig. 2.4. Response on the neural network [4].

The response of the system reacts as expected but in complex real life situations with different background the implementation of this kind is much more complicated being necessary also a "Topological Feature Estimator" since the algorithm for the LGMD model involves losing information about the shape of the object originating the collision [2].

2.3 Collision avoidance model based on a hybrid fly-locust visual system

One model of a bio-inspired neuronal obstacle avoidance system that can be used as a control for air vehicle was proposed in "The International Journal of Robotics Research" [62] and it will be next analyzed. The system is supposed to include components for course stabilization, altitude control and collision avoidance and is based on the fly-locust opto-motor. The model follows different processes performed in the biological layers of the fly and locust visual system (Fig2.5). The first stage in the visual processing appears at the level of the compound eye, acting like an acquisition system. Next, in the lamina where the luminance signal received by the photoreceptors is normalized using a logarithmic compression :

$$I_{photo} = k_i . log\left(k_j I_{input} + const\right) \quad (2.12)$$

where I_{photo} is the photoreceptor response, I_{input} the input luminance level and k_i, $k_{j,}$ const are scaling constants. Subsequently an edge enhancement is performed in the

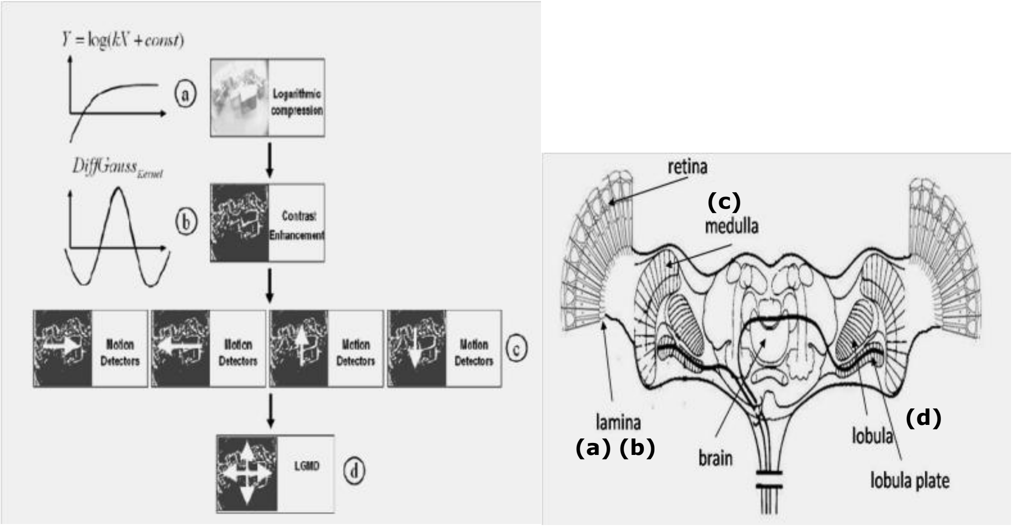


Fig. 2.5. Functional (left) and anatomical (right) structure of a prototypical insect visual system. Reproduced from [9], [62].

Lamina using using a centre/surround inhibition, a method similar to a difference of Gaussians based zero-crossing edge extraction [62]:

$$Edge_{image} = Input_{image} * DiffGauss_{Kernel} \quad (2.13)$$

where *Inputimage* is an input image, *DiffGaussKernel* a difference of Gaussians kernel and *Edgeimage* the resulting imag from the convolution operation.

$$DiffGauss_{Kernel} = f(\mu,\sigma 1) - f(\mu,\sigma 2) \quad (2.14)$$

$$f(\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma}} exp\left(\frac{-(x+\mu)^2}{2\sigma^2}\right) \quad (2.15)$$

with μ the mean value, and σ1 > σ2 fixed standard deviation values.

After contrast information is extracted using the process described above, the next process extracts the optic flow information relevant for flight stabilization, altitude control and collision avoidance, the top priority being an avoidance action.
These optic flow patterns are detected by the, so called, wide-field Horizontal and Vertical System neurons (HS and VS respectively) located in the Lobula plate layer (Hengstenberg 1982)."These cells are known to be motion sensitive and they respond maximally to a stimulus moving in a certain preferred direction whereas they show a decrease in the membrane potential due to stimuli moving in the opposite direction)"[62]. The reaction of these types of neurons result from the incorporation of the activity of local visual motion cells called Elementary Motion Detectors (EMDs) as it was mention earlier. The Reichardt correlation model is

applied at the pixel level between neighboring pixels (*Ia* and *Ib*) separated by a certain distance *D*. There are two branches, the null and preferred output, which are computed independently. If the is translating object from pixel *a* to *b* at speed v, the Reichardt correlation Rcorr(Ia, Ib) will be:

$$R_{corr}(I_a, I_b) = Out_{preferred}(I_a, I_b) - Out_{null}(I_a, I_b) \quad (2.16)$$

$$Out_{preferred}(I_a, I_b) = I_a(t-\delta) \cdot I_b(t) \quad (2.17)$$

$$Out_{null}(I_a, I_b) = I_a(t-\delta) \cdot I_b(t) \quad (2.18)$$

Knowing the speed v and the pixel separation D :

$$I_b(t) = I_a(t - D/v) \quad (2.19)$$

$$Out_{preferred}(I_a, I_a(t - D/v)) = I_a(t-\delta) \cdot I_a(t - D/v) \quad (2.20)$$

Out$_{preferred}$ and Out$_{null}$ are maximum when:

$$\frac{\partial}{\partial\delta} Out_{preferred} = 0 \text{, for } \delta = D/v \quad (2.21)$$

$$\frac{\partial}{\partial\delta} Out_{null} = 0 \text{, for } \delta = -D/v \quad (2.22)$$

It can be observed that the best response of the Reichardt detector is for a particular speed related to the delay. After computing the response difference of left and right, and up and down motion sensitive neurons, a decisional block generates an appropriate compensation motor command.

The model proposed assumes that directional motion information of the specific motion detectors of the lamina is used *a posteriori* to detect expanding. Also, the EMD responses found in the lamina are integrated by the LGMD in order to compute the variation of expansion from the visual field, so it will generate pulses when this integrated signal is over a specific threshold [60], [62].

3 Enhanced bio-inspired motion and obstacle detection system concepts

3.1 Elaborated motion detector based on EMD model

Based on the EMD cell next is presented a concept of an elaborated motion detector which can distinguish between an object moving towards the front direction of an object moving horizontally. In Fig.3.1 is the block diagram of the improved elementary motion detector [1], [60], [62].

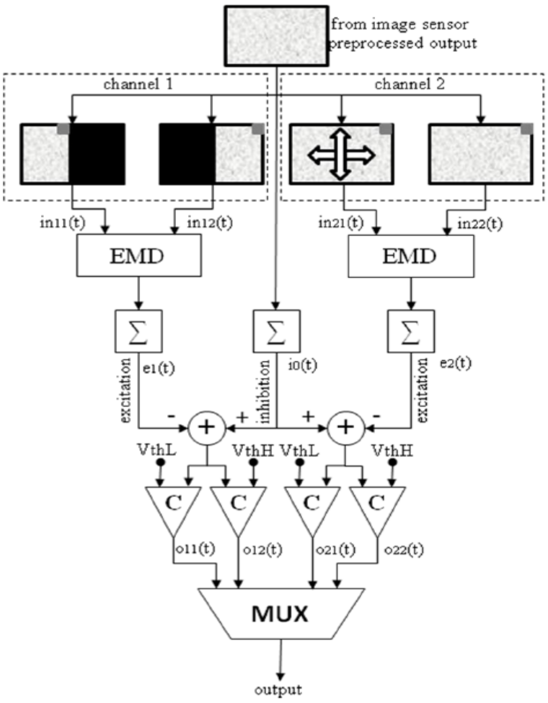


Fig. 3.1. The proposed motion detector block diagram [1].

The first step is in a preprocessing block where an edge detection algorithm to the received images from the sensor at pixel level is distributed at two channels where is computed differently [1]. In the channel 1, the image is divided in two identical parts, symmetrical by the vertical axes. The values of the pixels are computed by the EMD. The sum of all pixels resulted by the EMD's output is an excitation signal that contains the information about the sense of the motion in the horizontal line (motion from right to left or from left to right) [1].

A similar process is in the other channel, where the image is divided also in two parts, but in this case, one part remains unchanged and the other part enlarges the image by both axes. The sum of the correlated pixels received by de EMD's output between these parts is an excitation signal that shows the closeness or distance of the object that is in front of the sensor in order to distinguish between an object that is approaching the sensor or on contrary it is moving away.

In order to compute this kind of information, the excitation signals from both of the channels must be confronted with a reference signal that can be named as an inhibition signal. This reference signal is obtain from the summation of all pixels from the preprocessed image. The difference between the excitation and the inhibition signals is compared with an upper and a lower threshold (V_{thH} and V_{thL}) resulting the sense of the moving object direction by the case [1].

3.1.1 Simulation of the proposed motion detector

In order to evaluate the presented concept, the system was simulated using Matlab/Simulink tool. The input for the system were some recorded videos using a low resolution webcam (160x120 pixels) and the edge detection was made using Canny algorithm [1].

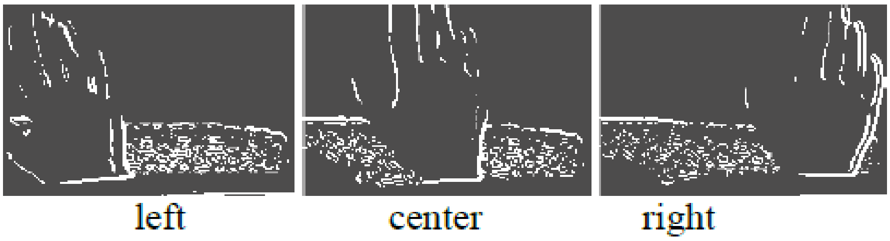


Fig. 3.2. Positions of the object in horizontal plane during the simulation [1].

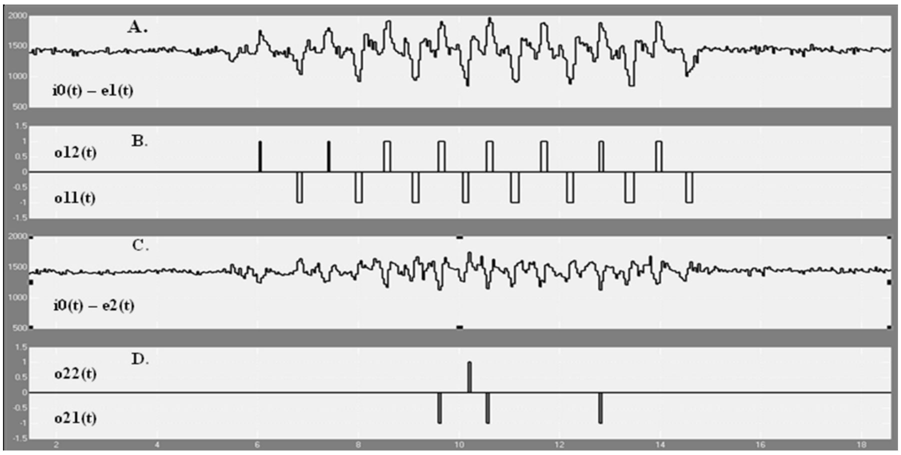


Fig. 3.3. Horizontal motion detection simulation results [1].
A. Channel-1 inhibition-excitation differential signal i0(t)-e1(t)
B. Channel-1 outputs : o11(t) , o12(t)
C. Channel-2 inhibition-excitation differential signal i0(t)-e2(t)
D. Channel-2 outputs : o21(t) , o22(t)

In Fig.3.2 are presented a few snapshots from the input video and Fig.3.3 there are waveforms representing the output of the computational blocks after running the simulation. In the first case a hand is moving in horizontal plane from right to left and from left to right with a rest time between successive movements. As shown in the diagram from Fig.3.3, the first channel respond more intense with a train of pulses with a sign that indicates the sense of the motion, in this case +1 was choose to indicate the right left movement and -1 the left-right movement. In Fig.3.4, the hand moves back and forth in front of the camera so now the channel 2 is more reactive [1]. Also for this case the sign of the pulse shows the sense of the motion, -1 when the object is approaching and +1 when is moving away. The opposite channel

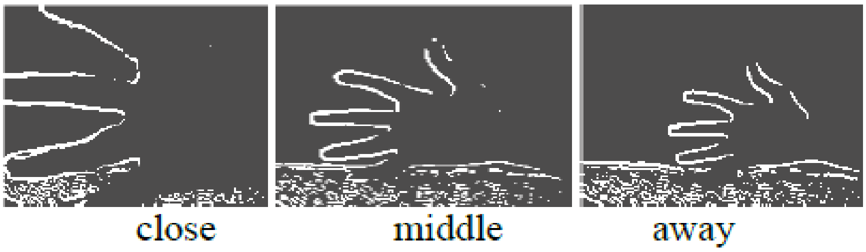


Fig. 3.4. Positions of the object in front of the sensor during the simulation [1].

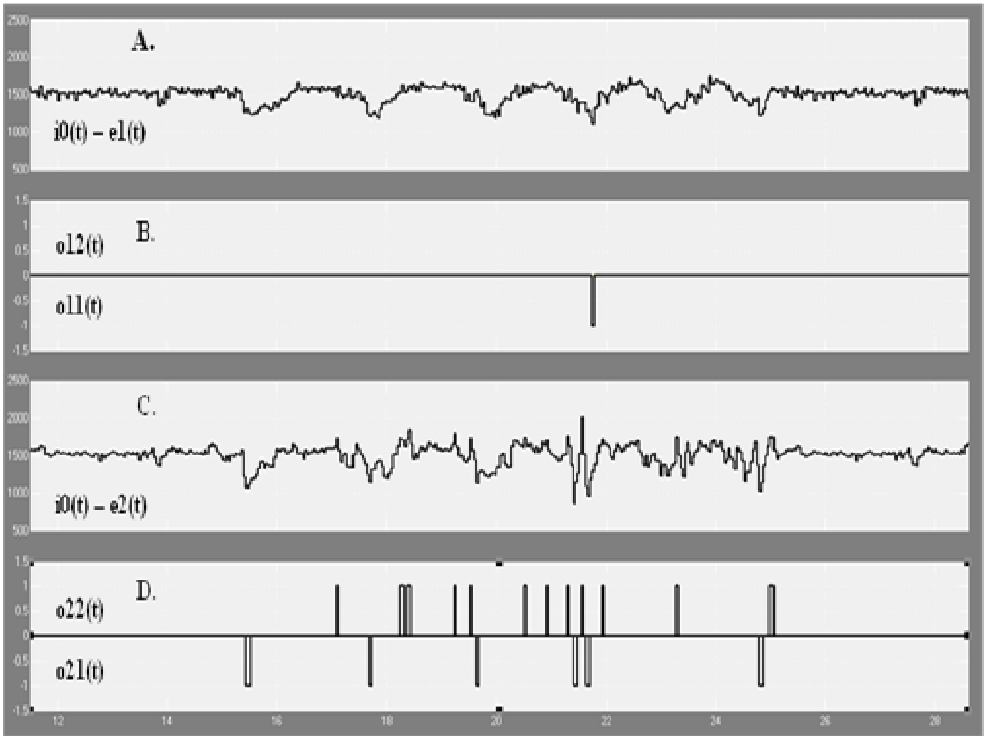


Fig. 3.5. Frontal motion detection simulation results [1].
A. Channel-1 inhibition-excitation differential signal i0(t)-e1(t)
B. Channel-1 outputs : o11(t) , o12(t)
C. Channel-2 inhibition-excitation differential signal i0(t)-e2(t)
D. Channel-2 outputs : o21(t) , o22(t)

In a similar way in the case of a moving object back and forth in front of the sensor shown in Fig.3.4, pulses from Fig.3.5, diagram D are generated accordingly to the inhibition-excitation differential signal from Fig.3.5, diagram C, that goes below a lower limit V_{thL}, or exceeds a upper limit V_{thH}. Also, the opposite channel will generate some spikes due to the correlation process [1].

3.2 Enhanced Bio-inspired Collision Avoidance System Concept with Fuzzy Logic Decision

Humans always used nature as a source of inspiration for creating tools that can be used to achieve different tasks in everyday life. Implementing an electronic device that can replace traditional tools like white cane and training dogs that are used to guide people with visual disabilities is quite a challenge. The insect visual system is a great source of inspiration for such devices because even if insects are considered to be more inferior in complexity in comparison with the vertebrates, they possess a simple but yet effective way to orientate in space and accomplish complex tasks such as: obstacle avoidance, escape from predators, landing [9],[61].

When it comes to implement an artificial system whose role is to detect obstacles that intersect the trajectory of motion, must be explained the biological mechanism that stand at the base of this process. An important condition in order appear a collision with an obstacle is that we must have motion, so the first is to detect the relative motion of the obstacles from the visual field. Of course, not all the objects that are moving around us, represent a real danger for collision, so next step is to determine which one of them are with a potential risk to collide with. There is sometimes a false impression that in order to be able to detect an obstacle it is necessary to know the exact distance between the subject and the obstacle. For example, let's take the situation when a person wants to cross a street and in the same time a car is approaching [9]. That individual can calculate if he can cross the street before the car comes and strikes him but not by calculating the actual range from him to the car or the actual speed of the car, just simply by the fact that motion of the car creates a visual stimuli to the brain which based on previous experiences, can decide if the time is sufficient to cross the street safely. This is available not only for humans, but for all other species that use a visual system, including the insects which are the main inspiration for this research [9].

3.2.1 Proposed Bio-Inspired Obstacle Detection System Concept with Fuzzy Logic Decision

The role of the bio-inspired obstacle detection system is to determine if an object is approaching the sensor or the opposite when the sensor is approaching an object that can be either static or in motion. Most of the time in order to avoid collision with an obstacle it is not necessary to know a lot of information which the human brain processes based on the visual senses [9]. For example: it is not necessary to know if an object has specific color, or what shape has, neither what are the exact dimensions, or what is the structure of the material.

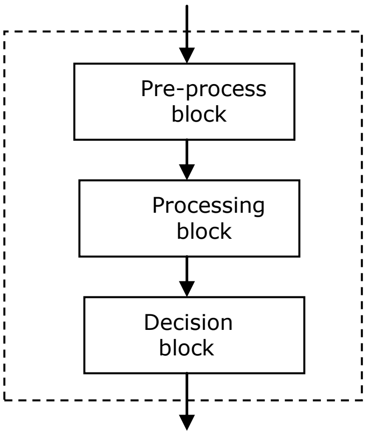


Fig. 3.6. Main block diagram of the proposed obstacle avoidance [9].

The proposed bio-inspired obstacle detection system, according to [9] is made by 3 parts: the pre-processing block, the processing block and the decision block. In the pre-processing part the output of the visual sensor is transformed to grayscale and then an edge detection algorithm is applied. The reason for applying edge detection to the images received is to eliminate some of the redundant information and by that to increase the processing speed. By applying this edge detection is obtained a matrix of connected curves that encompass objects indicating in the same time the discontinuities in the surface orientation, but this comes with some disadvantage here because sometimes not only the redundant information is eliminated but also some of the important information because edges from the image are retrieved with fragmentation, meaning that the edge curves are not connected, and so the missing edge segments will not correspond to the object that could be on the collision course. There is different kind of edges, "viewpoint independent" types which reflects the object surface markings and also if some changes occur in the orientation of the surface, and "viewpoint dependent" where the edges are changing according to the relation between the viewer and the background [9].

In the processing part the computation is accomplished at pixel level and represents the part in which is extracted the information that contains a possible collision with the obstacle that interferes in the trajectory of motion. The main insects' travel environment is the air, so usually obstacles which they may encounter are in the air. The concept proposed here is to guide people with visual disabilities, this is way has to be adapted for land traveling. Extracting from the visual field the right information play an important role, because if it is extracted only the center of the visual field, there are situations when the background contains a lot of edges and even if the object is far away, the system will generate collision signals [9]. On the other hand if it is extracted for further processing only the walking path area, it may happen that the center of the visual field itself represents an obstacle and the collision signals will not be generated in time. So, if both parts are taken into account for processing the final decision will be more appropriate with reality [9].

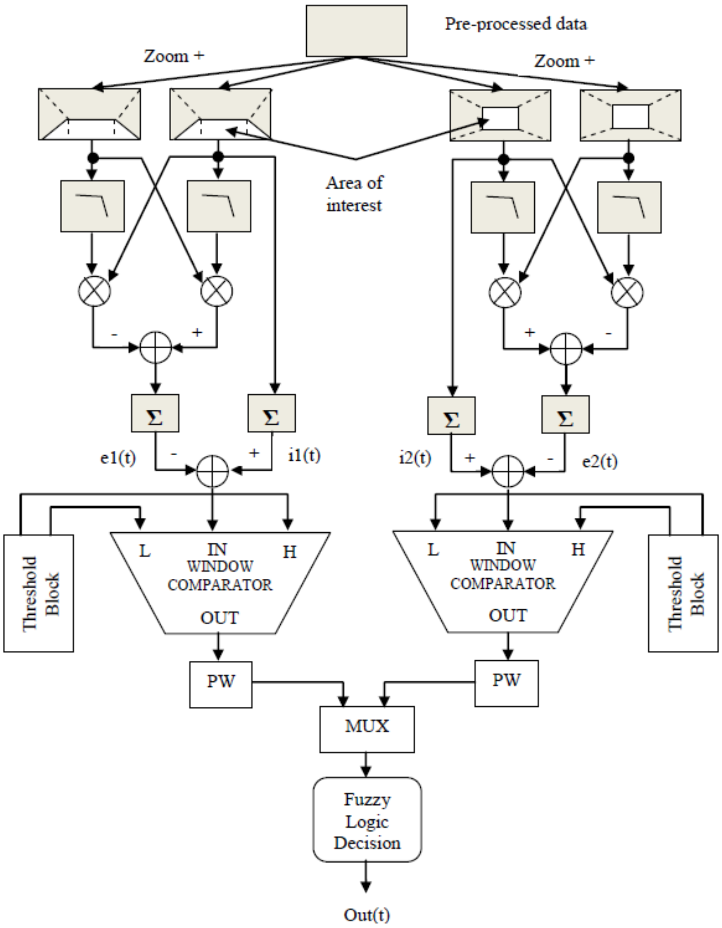


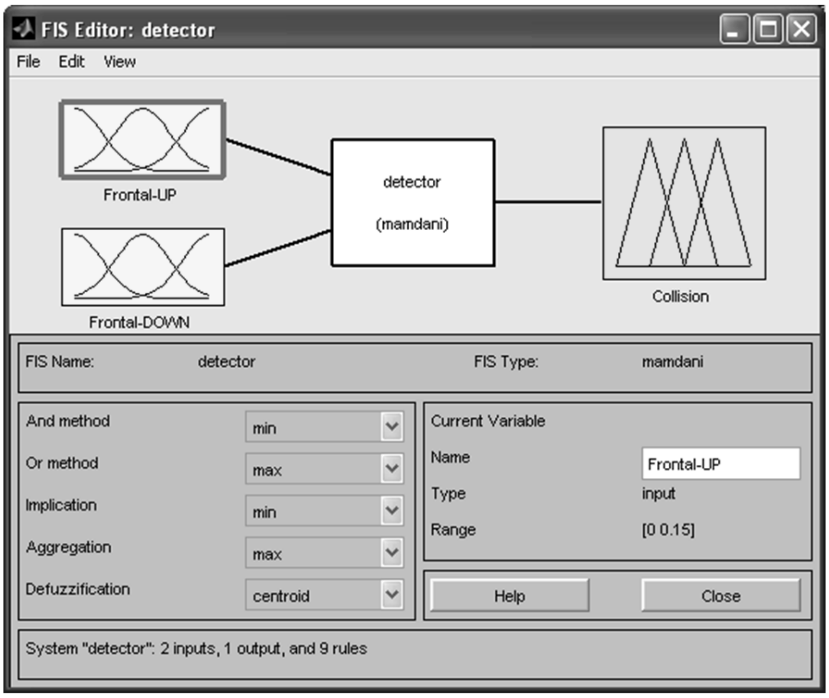
Fig. 3.7. Detailed block diagram of the enhanced bio-inspired collision avoidance system [9].

The processing part is divided in two channels were the information is computed differently. The first channel process the information regarding the obstacles that interfere in the walking path area. This is done by correlating continuously the current frame received with the enlarged frame by both axes of the same image, using the "Elementary Motion Detector" [1], [9], [62]. The pixels from the output frame of the EMD block are summed to generate an excitation signal (e1(t)) that will be compared against a reference (inhibition) signal provided by the summation of all pixels from the non-magnified walking path area (i1(t)).

The difference between the excitation and the inhibition signal (e1(t) – i1(t)) changes continuously the mean value in accordance with the environmental light, so the solution to use fixed thresholds for the window comparator is not a good idea. To eliminate this issue, the "Threshold block" extracts the mean value from the [e1(t) – i1(t)] signal and uses it as reference to generate the threshold for the window comparator. In this case the lower threshold for the window comparator is obtained from the sum between the mean value of the excitation and the inhibition signal difference, plus a threshold value, while the upper threshold for the window comparator is obtained from the difference between the mean value of the excitation and the inhibition signal difference from which is subtracted a threshold value [9].

The second channel computes the signal received from the pre-processing block in a similar manner, except that in this case the interest area is the center of the visual field. The process part is the same as in the case of the walking path area.

The window comparators will generate a train of pulses with different width according to the closeness or distance of the objects from the trajectory of motion which will be evaluated by a fuzzy logic controller in order to appreciate the danger of a possible collision with an obstacle. The aggregation method used was "maximum" and the "defuzzification" process was chose the "centroid" calculation method type.



FIS Name: detector

FIS Type: mamdani

And method: min

Or method: max

Implication: min

Aggregation: max

Defuzzification: centroid

Current Variable

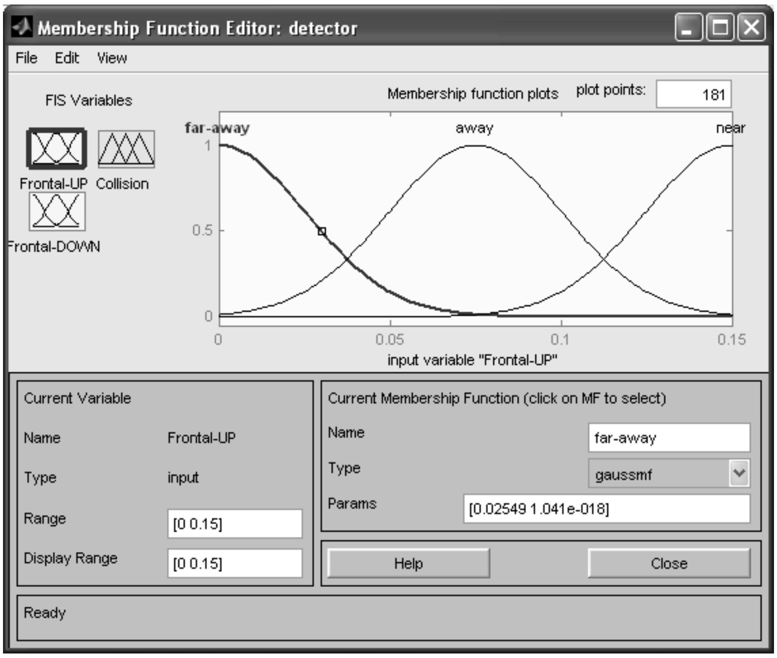
Name: Frontal-UP

Type: input

Range: [0 0.15]

System "detector": 2 inputs, 1 output, and 9 rules

Fig. 3.8. Fuzzy Inference System (FIS) Matlab GUI editor.



plot points: 181

Current Variable

Name Frontal-UP

Type input

Range [0 0.15]

Display Range [0 0.15]

Current Membership Function (click on MF to select)

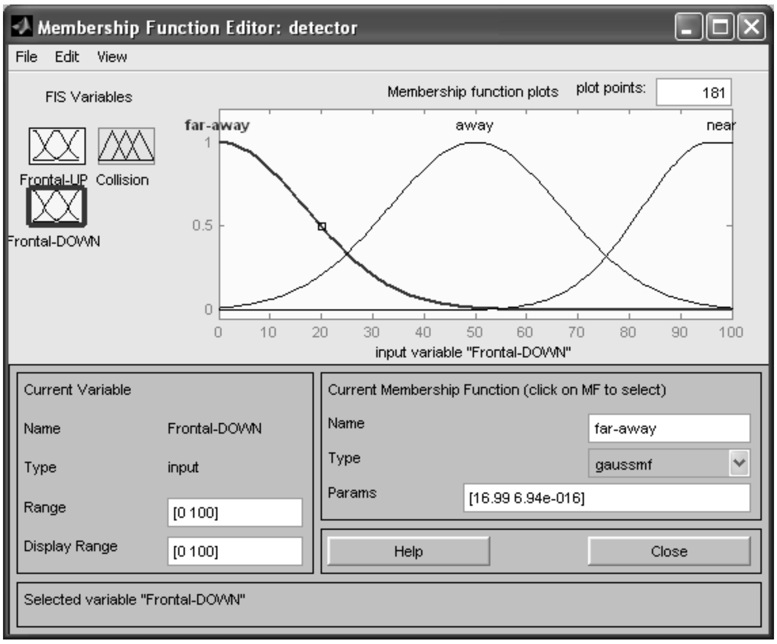
Name far-away

Type gaussmf

Params [0.02549 1.041e-018]

Ready

Fig. 3.9. Gaussian type membership function for the Frontal-UP channel.



plot points: 181

Current Variable

Name Frontal-DOWN

Type input

Range [0 100]

Display Range [0 100]

Current Membership Function (click on MF to select)

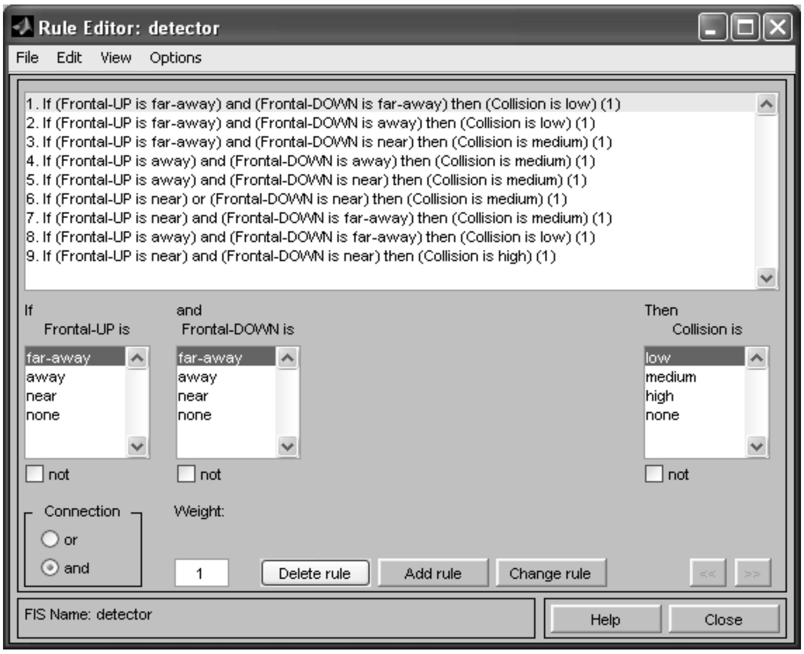
Name far-away

Type gaussmf

Params [16.99 6.94e-016]

Selected variable "Frontal-DOWN"

Fig. 3.10. Gaussian type membership function for the Frontal-DOWN channel.



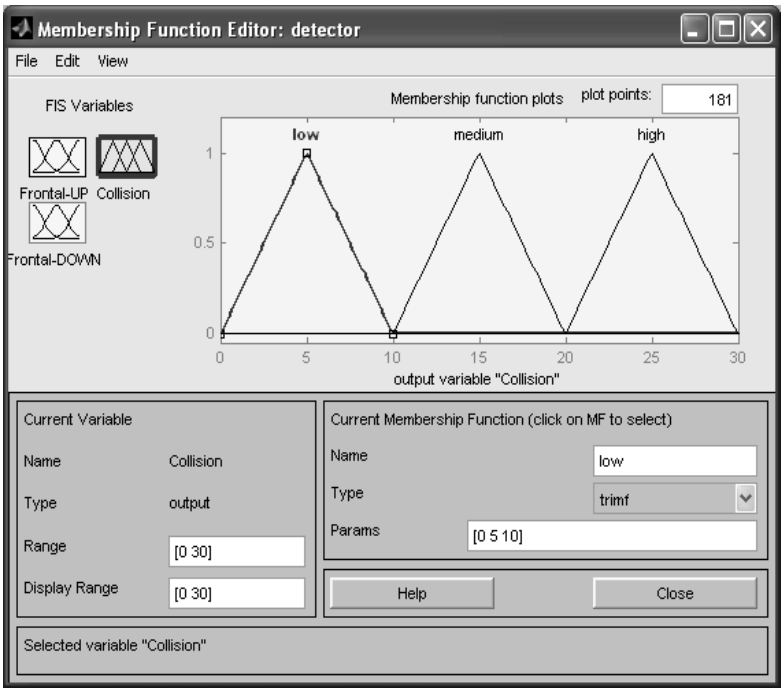
1. If (Frontal-UP is far-away) and (Frontal-DOWN is far-away) then (Collision is low) (1)
2. If (Frontal-UP is far-away) and (Frontal-DOWN is away) then (Collision is low) (1)
3. If (Frontal-UP is far-away) and (Frontal-DOWN is near) then (Collision is medium) (1)
4. If (Frontal-UP is away) and (Frontal-DOWN is away) then (Collision is medium) (1)
5. If (Frontal-UP is away) and (Frontal-DOWN is near) then (Collision is medium) (1)
6. If (Frontal-UP is near) or (Frontal-DOWN is near) then (Collision is medium) (1)
7. If (Frontal-UP is near) and (Frontal-DOWN is far-away) then (Collision is medium) (1)
8. If (Frontal-UP is away) and (Frontal-DOWN is far-away) then (Collision is low) (1)
9. If (Frontal-UP is near) and (Frontal-DOWN is near) then (Collision is high) (1)

Connection
○ or
● and

Weight: 1

FIS Name: detector

Fig. 3.11. Description of the implemented rules of the fuzzy controller.



plot points: 181

Current Variable

Name Collision

Type output

Range [0 30]

Display Range [0 30]

Current Membership Function (click on MF to select)

Name low

Type trimf

Params [0 5 10]

Selected variable "Collision"

Fig. 3.12. Triangle function type for the output variable of the fuzzy controller.

3.2.2 Simulation results

The concept for the proposed bio-inspired obstacle detection was simulated using the Matlab\Simulink software. For the input of the system were used recorded videos from the real environment with a camera carried by a person that is moving on the street [9]. The simulated situations when the subject is moving in a city street and the second one in an alley of a park [9]. The closeness or the distance between the subject and the object is given by the fuzzy logic controller of the decision block and evaluated on a scale of 1 to 10.

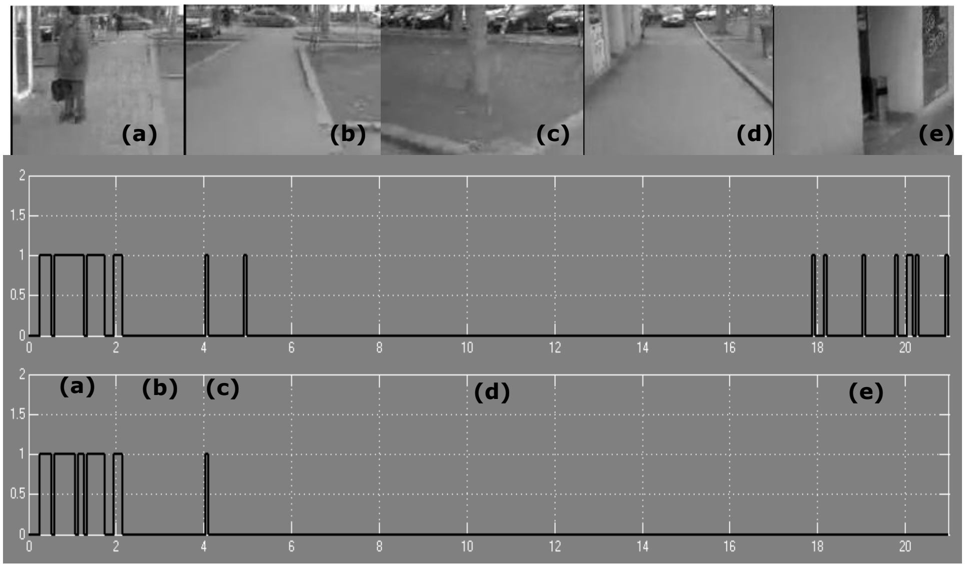


Fig. 3.13. Snapshots taken from the subject motion along the street and the response of the window comparators [9].
(1 – center of the visual field; 2- walkin path area)

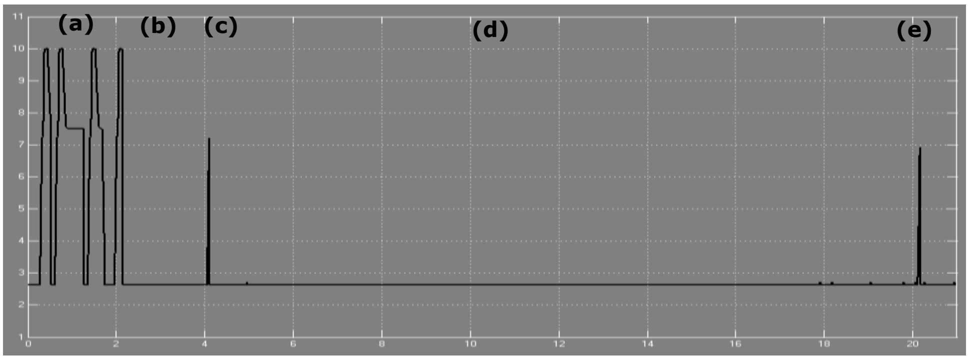


Fig. 3.14. Response of the Fuzzy logic controller with the corresponding situations from the snapshots in the case of the subject's motion along the street) [9].

In Fig.3.13 is the first situation where the person is moving on the street and interacts with different objects, which is reflected by the frames from "A" to "E" with their correspondences of the response from the two window comparators. The closeness or distance between the person and the objects that appear in the trajectory of motion is reflected in Fig.3.14 by the output of the decision block that generates signals with different levels according to the width of the pulses from the window comparators. As can be seen in the frame "A" for example, the person that stands in the trajectory of motion is much closer to the subject than the tree from frame "C", and this is reflected by a higher level of response from the decision block. Similar situations are presented in Fig.3.15 and Fig.3.16 case where the person is moving along the park's alley. Same in here, when the person is closer to an object the system will respond with a higher signal.

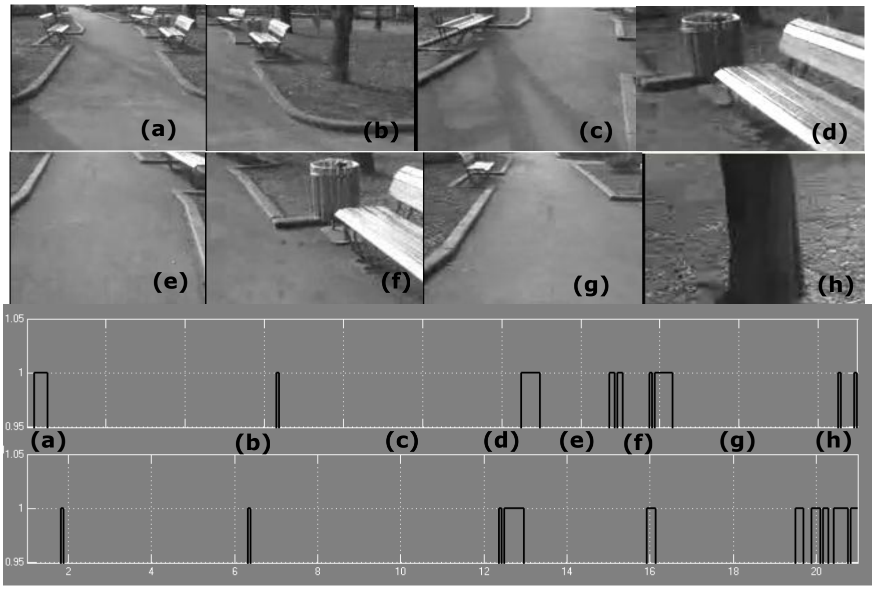


Fig. 3.15. Snapshots taken from the subject motion along the park's alley and the response of the window comparators [9].
(1 – center of the visual field; 2- walking path area)

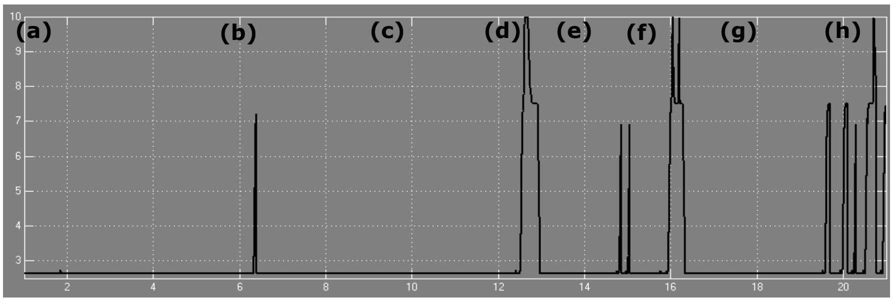


Fig. 3.16. Response of the Fuzzy logic controller with the corresponding situations from the snapshots in the case of the subject's motion along the park's all [9]

4 Implementation solutions for the bio-inspired obstacle detection system

When it comes to implement a solution for a computational model, sometimes it is difficult to choose between a custom chip design that fulfill the requirements and an embedded system that does the same functions. Now days CMOS technology has reached a mature state having the advantage of lower power consumption, but it still has some issues with thermal noise generated by the silicon substrates which limits the linear operational for analog devices. Still, sometimes choosing an analog design over a digital one has major advantages. For example, an arithmetic circuit implemented in a digital technology requires hundreds of transistors, on contrary an analog implementation of the same function requires a few transistors only. Even in these conditions, generally is still preferred a digital approach over an analog one and this can be explained by the "degree of difficulty" that sometimes is higher for an analog design.

A few examples where analog circuits are much competitive can be given: adaptive equalization for mobile communications [66], [75] equalizers for modern magnetic storage channels [67], [75] and adaptive noise cancelation for telephone systems [68], [75] and of course in bio-inspired structures where analog circuits are approaching the real systems. In addition, new applications continue to appear where the use of analog front ends provide the required performance with less power and area than the digital counterparts [69], [70], [71], [75]. The main disadvantage of analog design for signal processing applications is in the *degree of design difficulty*.

4.1 Concept for an analog VLSI obstacle detection system

When an animal moves through the environment, images of the outside world move across its eyes and the objects that are approached are perceived as growing larger and objects that are left behind became smaller. Also regarding the speed of the moving objects, they are perceived differently, the ones nearby are faster than images of distant objects. If the animal executes a rotational motion, the entire visual field moves across its eyes at a rate that is independent of object distance. "Motion parallax information is immune to camouflage that can defeat even the most sophisticated static pattern recognition scheme when object and background have similar textures" [82].

A biological visual system is a high parallel structure that computes in a different way from traditional digital computers. Using the advantage of VLSI (Very Large Scale Integration) technology, it can be done silicon chips with millions of transistors that emulate functions of the biological neuro-structures with power efficiency very convenient. Actual technology permits to implement photodetectors in the same substrate with the computational elements. "Many different analog VLSI imagers with integrated motion detectors have been built in the past decade, and functionality is slowly increasing." (Delbruck, 1993; Etienne-Cummings and Van der

Spiegel, 1996; Sarpeshkar et al., 1996; Moini et al., 1997; Higgins et al., in press) [82].

4.1.1 Bio-inspired Adaptive Logarithmic Photoreceptor

Photoreceptors used in visual processing in comparison with the imaging are different because sampled imagers are designed to go with television transmission, and they have to reproduce exactly the visual scene with no additional computation.

A receptor that mimics a biological sensor has to have a logarithmic response relative to illumination and a simple way to obtain it is to use a photodiode structure.

An example of a logarithmic receptor can be implemented between a lightly doped substrate and heavy doped source–drain diffusion. The electrons released in the p− substrate diffuses into the junction and then back into the n++ region. In a similar way the holes formed in the n++ region grabbed back into the p− region. After this process the n++ region becomes negatively charged. The forward current in the junction is exponential so the voltage in the n++ region is logarithmic but being under the substrate voltage the signal can't be used practically.

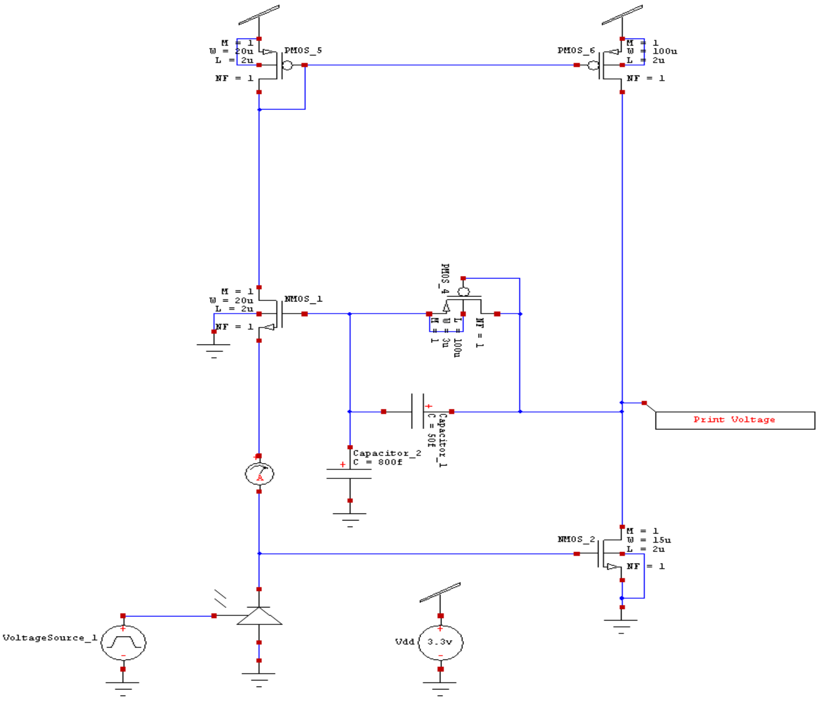


Fig. 4.1. Self-biasing adaptive logarithmic photoreceptor.

In Fig. 4.1. It is presented a self-biasing photoreceptor circuit according to [83], [84].

The feedback loop operates as fallow: at low frequencies the adaptive element acts like a short-circuit providing a unity gain and at high frequencies the two capacitors act like a divider resulting in a higher gain for transient signals, a behavior that is similar with the biological receptor [83], [84], [87].

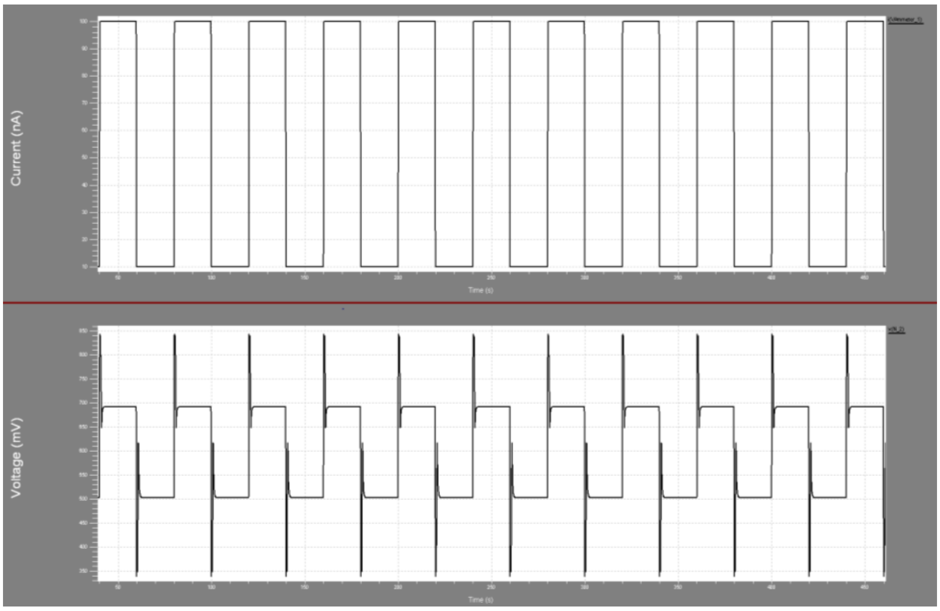


Fig. 4.2. The current generated by the photodiode. The response of the adaptive photoreceptor from Fig.4.1.

Some disadvantages of the circuit is that the value for the capacitor C_b is almost critical and if it is too small the circuit can oscillate and if it is too big will reduce the speed response for a step change. Another disadvantage is that the DC gain is higher for this type of structure because of the mirror Mb-Mp that gives the bias current for the forward amplifier, a current that is proportional with the generated photocurrent, also because of the mirror the output impedance is higher than in the case when is used a statically-biased photoreceptor and it has to take into account when the photoreceptor will drive other computational circuit [83], [84], [87].

A new structure for the bio-inspired adaptive logarithmic photoreceptor is proposed in Fig.4.3. The new concept wants to reduce the DC gain and to increase the gain for the transient signals. For this an additional current mirror, a current comparator and 2-to-1 Multiplexer were used. When the photodiode generates a higher current the forward amplifier will be biased will a lower current from a current source then in the schematic from Fig.4.1 reducing this way the DC gain.

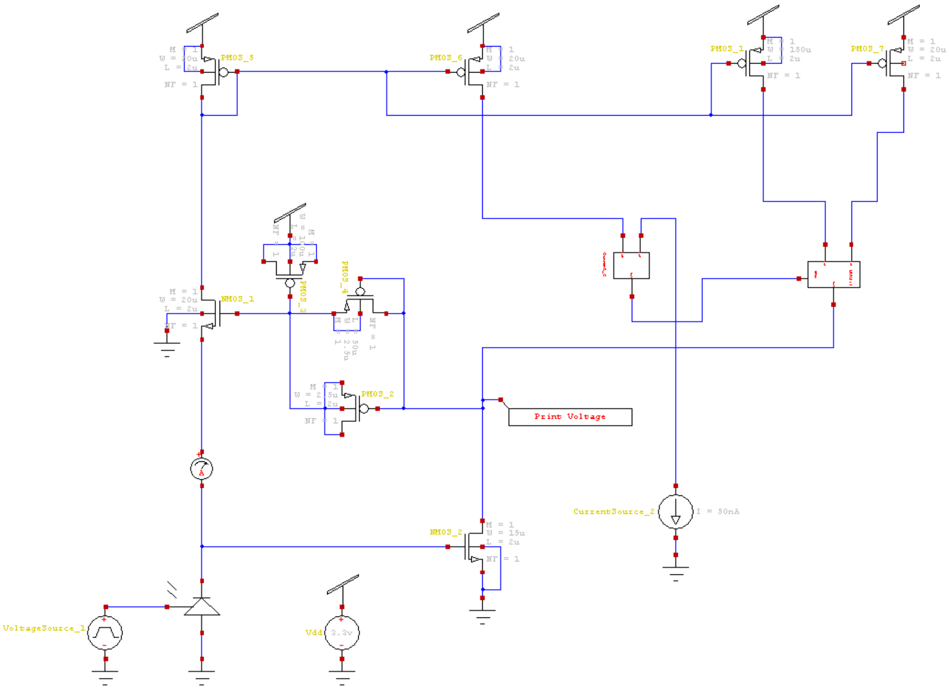


Fig. 4.3. Proposed bio-inspired adaptive logarithmic photoreceptor

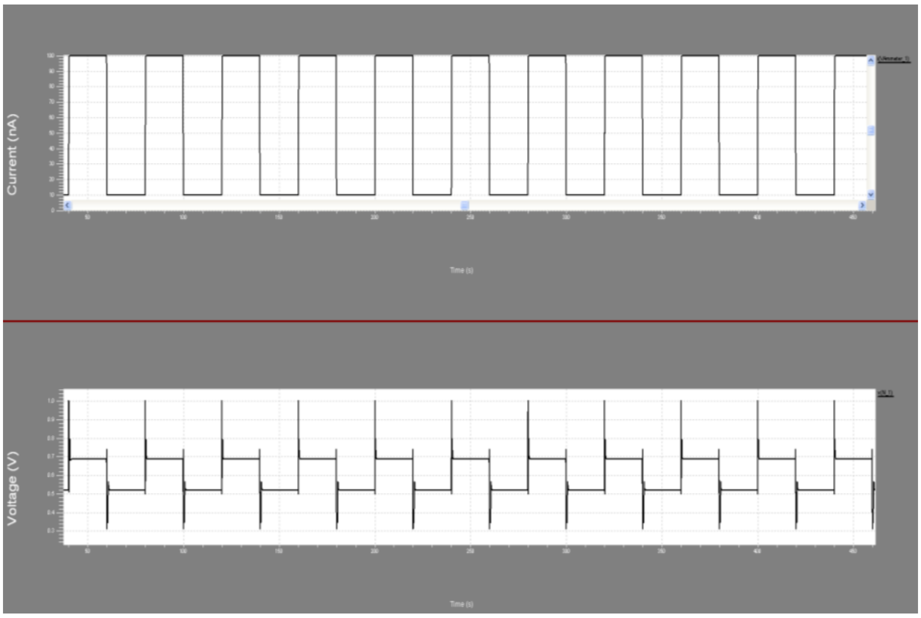


Fig. 4.4. Simulation of the proposed bio-inspired adaptive logarithmic photoreceptor. The current generated by the photodiode. The response of the adaptive photoreceptor.

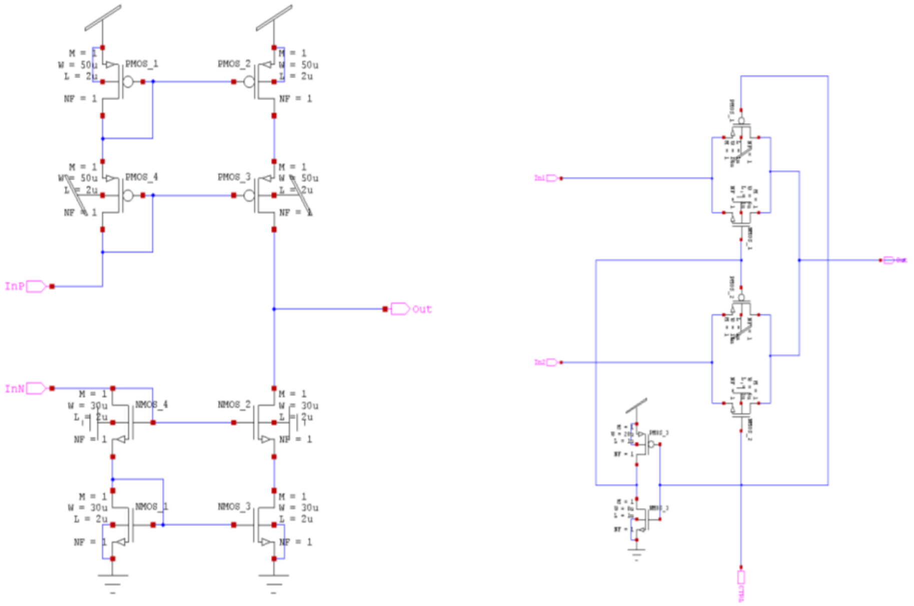


Fig. 4.5. (a) CMOS current comparator. (b) CMOS selector with transmission gates.

4.1.2 Half-Gaussian CMOS analog circuit function

According with the biological behavior of the insect visual system, the response of the adaptive logarithmic photoreceptor must be passed through a circuit with a Gaussian type response. Since the output of the previous proposed photoreceptor has only positive values it will be used only a half Gaussian circuit type similar with the one proposed in [95].

A definition for the gaussian function can be given by the following formula:

$$y = Ae^{-\frac{x^2}{2\sigma^2}} \qquad (4.1)$$

where "x" represents the input and "y" represents the output, "A" and "σ" are parameters to adjust the amplitude, respectively the width of the Gaussian function.

The function is constructed in Fig.XX in the following way: the current mirror formesd with transistor NMOS_3 and NMOS_4 uses a current reference low enough to put these transistors in weak inversion mode, while transistors NMOS_1 and NMOS_2 work in their linear region as variable resistors [85], [86], [95].

$$I_{R=}I_S e^{\frac{V1}{nu_T}}\left(e^{-\frac{V_{S1}}{u_T}} - e^{-\frac{V_{D1}}{u_T}}\right) \qquad (4.2)$$

$$I_o = I_S e^{\frac{V_{G2}}{nu_T}} \left(e^{-\frac{V_{S2}}{u_T}} - e^{-\frac{V_{D2}}{u_T}} \right) \quad (4.3)$$

$$u_T = \frac{KT}{q} \quad (4.4)$$

"n" represents the slope factor and "I_s" is a current specific with the technology. Because of the operating region of the transistors, $V_{D1} >> u_T$ and $V_{D2} >> u_T$ and also $V_{G1}=V_{G2}$ results that :

$$I_R \cong I_S e^{\frac{V_{G1}}{nu_T}} e^{-\frac{V_{S1}}{u_T}} \quad (4.5)$$

$$I_o \cong I_S e^{\frac{V_{G1}}{nu_T}} e^{-\frac{V_{S2}}{u_T}} \quad (4.6)$$

From equations (4.5) and (4.6) results that :

$$I_o = I_R e^{\left(\frac{R_1 I_R - R_2 I_o - R_2 I_1}{u_T}\right)} \quad (4.7)$$

Because $I_1 > I_R$ and $I_o < I_R$ then :

$$I_o \cong I_R e^{-\frac{R_2 I_1}{u_T}} = I_R e^{-\frac{V_{S2}}{u_T}} \quad (4.8)$$

The drain current through the variable resistors and their conductance can be expressed as:

$$I_D = \beta (V_D - V_S) \left[V_C - V_{TN} - \frac{n}{2} (V_D - V_S) \right] \quad (4.9)$$

$$g_{ds3,4} = \frac{\partial I_{D3,4}}{\partial V_{D3,4}} \cong \beta_{3,4} (V_C - V_{TN3,4}) \quad (4.10)$$

and with the condition that

$$V_C - V_{TN} \geq V_{D3}, V_{D4} \quad (4.11)$$

The current through transistor NMOS_5 works in saturation:

$$I_1 = \frac{\beta_5}{2n} (V_{IN} - |V_{TP}|)^2 \quad (4.12)$$

So, after combining equations (4.8) and (4.12) , the transfer function becomes:

$$I_o = I_R e^{-\frac{I_1}{u_T g_{ds4}}} = I_R e^{-\frac{\beta_5 \left(V_{in} - |V_{TP}|\right)^2}{2nu_T \beta_4 \left(V_C - V_{TN}\right)}} \quad (4.13)$$

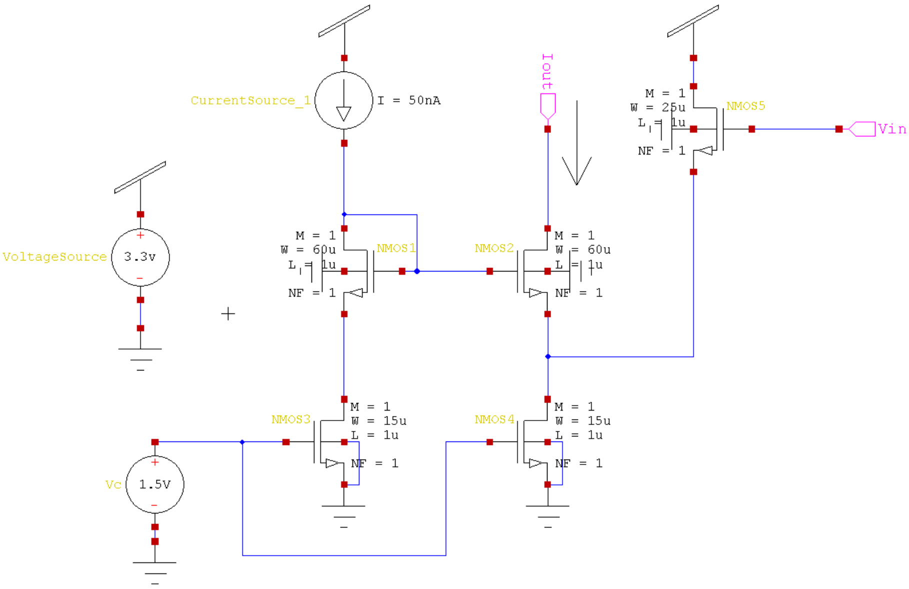


Fig. 4.6. Half-Gaussian type function using CMOS (S-edit implementation)

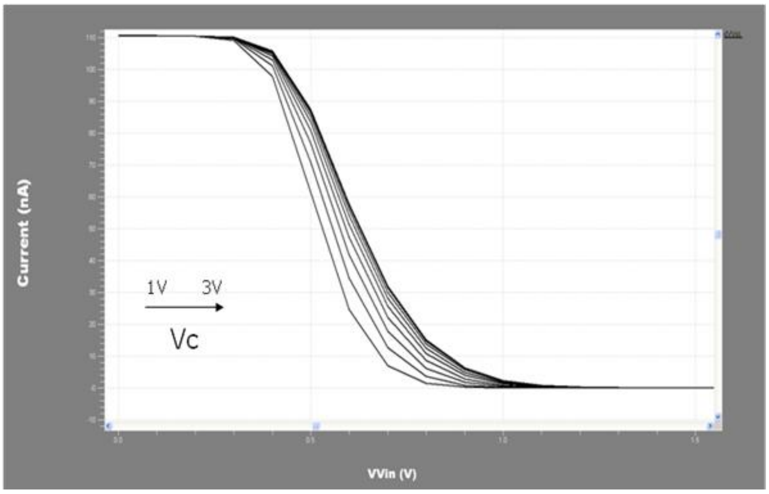


Fig. 4.7. Response of the half Gaussian type circuit for a Vc (voltage control) variation between 1V and 3V.

4.1.3 Complete schematic for a bio-inspired adaptive logarithmic photoreceptor with Gaussian response

The complete schematic for the photoreceptor is presented in Fig.4.8 and is composed by the adaptive logarithmic circuit, the half-gaussian circuit and a current-to-voltage convertor [98] since the output of the Gaussian circuit is a current.

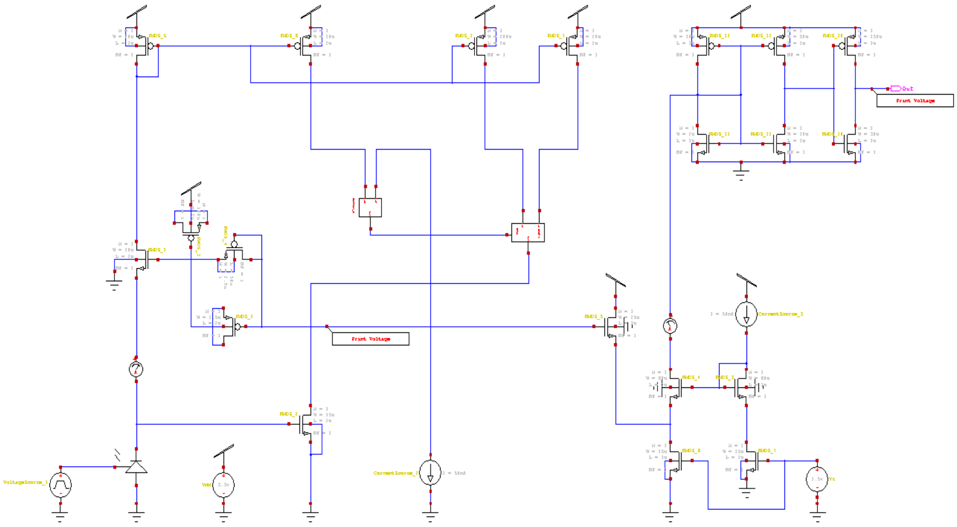


Fig.4.8. Enhanced bio-inspired adaptive logarithmic photoreceptor with Gaussian type response and voltage mode output

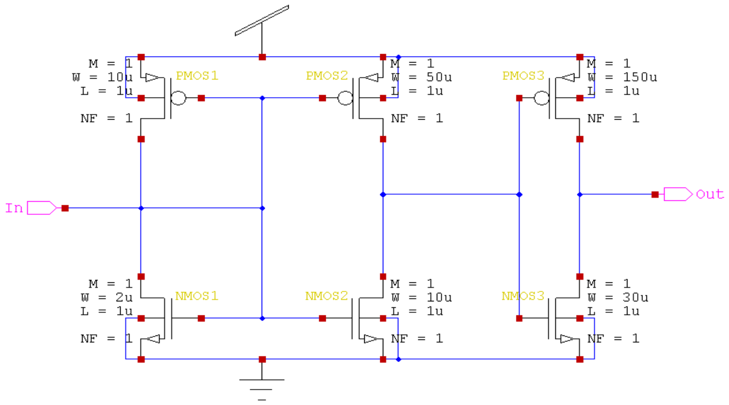


Fig. 4.9. Current-to-Voltage converter schematic (S-Edit Implementation)

Based on [98] it was used a current-to-voltage converter for currents raging in nA. In this case the variation of the common gate from the four transistors is small enough not to drive transistors PMOS2 and NMOS2 (Fig.4.9) out of the saturation region but it can modify currents through them, so NMOS2 and PMOS2 transistors tend to produce a difference of current that is equal with the input current.

Because there is no path for the difference of current to flow out, this current will change the draind-source voltage of these transistors and with an output resistance of tens or even hundreds of Mega ohms given by $r_{DS\text{-}NMOS2}$ || $r_{DS\text{-}PMOS2}$ the output voltage is obtained with a very high and almost constant gain.

The circuit from Fig.4.8 was simulated supposing that the photodiode produces a variation of a current ranging between 10 nA and 100 nA, the results being showed in Fig.4.10.

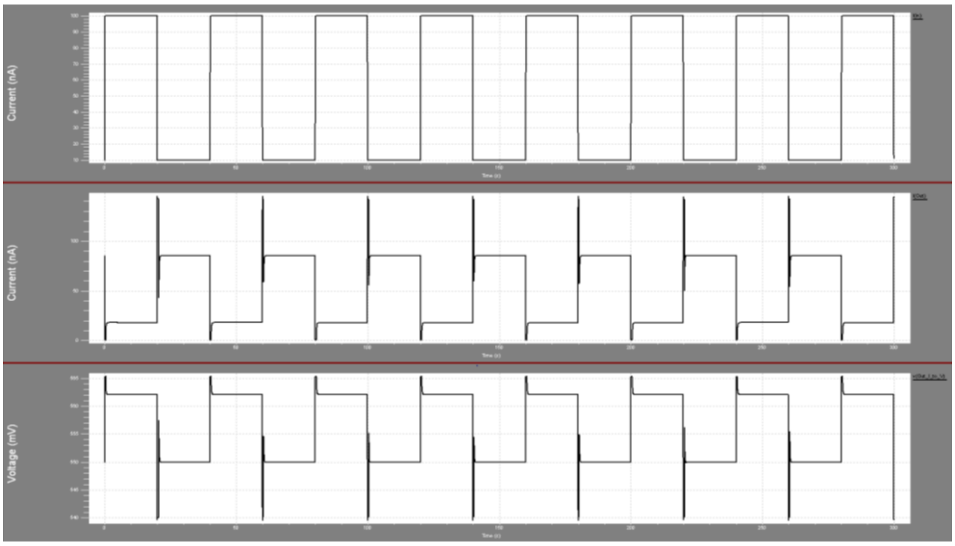


Fig. 4.10. Simulation of the schematic from Fig. 4.9. The current generated by the photodiode. The output current of the the Gaussian circuit. The output voltage after conversion.

4.1.4 Voltage-mode computational analog CMOS arithmetic circuits

Based on the behavior of the MOS transistor in the saturation region according with [75] are presented a few circuits that implement the basic arithmetic circuits and will be used in the processing of signals received from the photoreceptors.

In Fig. 4.11 is presented the schematics for the adder circuit in which there are 2 two single input voltages and a single output voltage.

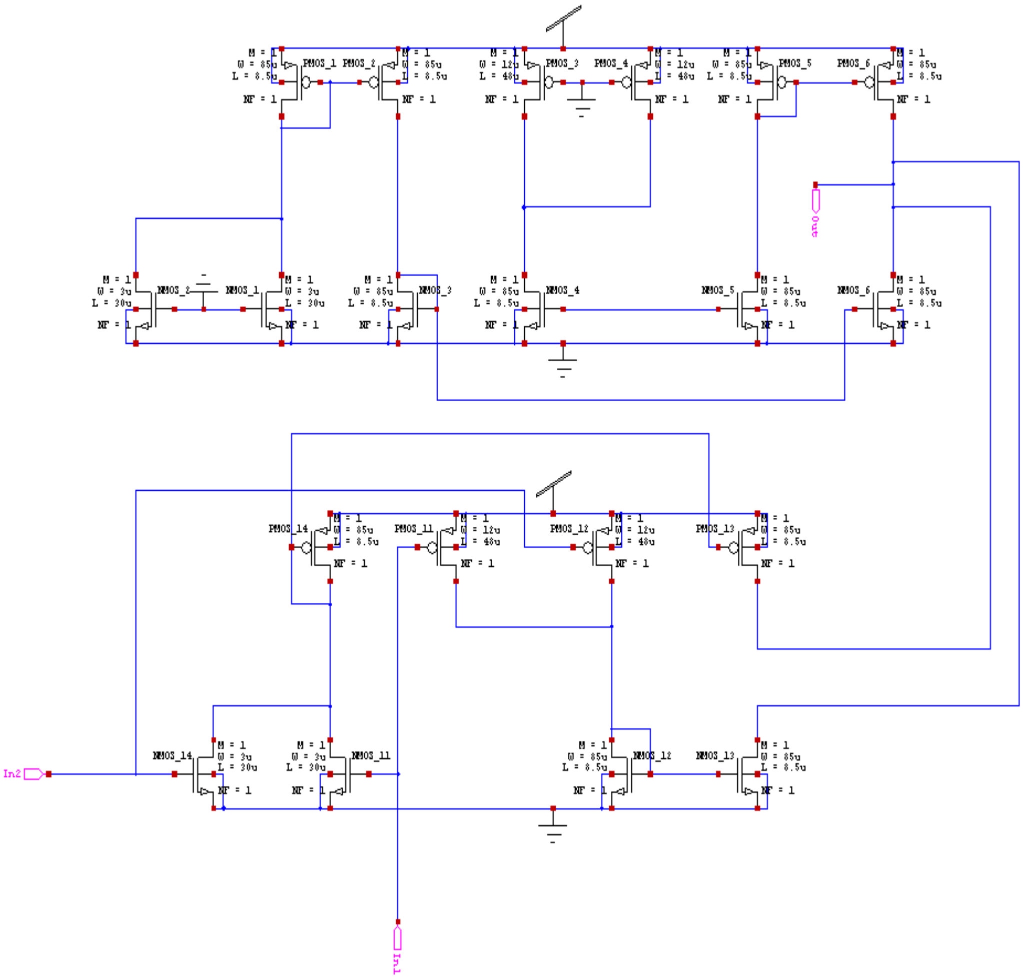


Fig. 4.11. Schematics for a CMOS analog Adder circuit (S-Edit implementation)

$$I_p = I_{14} + I_{16} = I_{21} + I_{23} + I_7 + I_9 \quad (4.14)$$

$$I_n = I_{15} + I_{17} = I_2 + I_3 + I_{18} + I_{20} \quad (4.15)$$

Knowing that in saturation region the equation for the drain current is :

$$I_{PMOS} = (1/2)\beta_P(V_{SG} - |V_{tp}|)^2 \quad (4.16)$$

$$I_{NMOS} = (1/2)\beta_N(V_{GS} - V_{tn})^2 \quad (4.17)$$

After simple calculations, the equation (4.14) and (4.15) becomes:

$$I_p = (1/2)\beta_n(V_1^2 + 2V_1C_n + C_n^2 + V_2^2 + 2V_2C_n + C_n^2) + \beta_p C_p^2 \qquad (4.18)$$

$$I_n = (1/2)\beta_p(V_1^2 - 2V_1C_p + C_p^2 + V_2^2 - 2V_2C_p + C_p^2) + \beta_n C_n^2 \qquad (4.19)$$

So the output current will be:

$$I_{out} = I_p - I_n = (1/2)(\beta_n - \beta_p)(V_1^2 + V_2^2) + (C_n\beta_n + C_p\beta_p)(V_1 + V_2) \qquad (4.20)$$

If it is considered that

$$\beta_n \approx \beta_p \qquad (4.21)$$

then:

$$I_{out} = (C_n\beta_n + C_p\beta_p)(V_1 + V_2) \qquad (4.22)$$

The output voltage for the Adder circuit with R$_{out}$ resistance will be:

$$\begin{aligned} V_{out} &= I_{out}R_{out} = \\ &= R_{out}(C_n\beta_n + C_p\beta_p)(V_1 + V_2) \\ &= K(V_1 + V_2) \end{aligned} \qquad (4.23)$$

with:

$$K = R_{out}(C_n\beta_n + C_p\beta_p) \qquad (4.24)$$

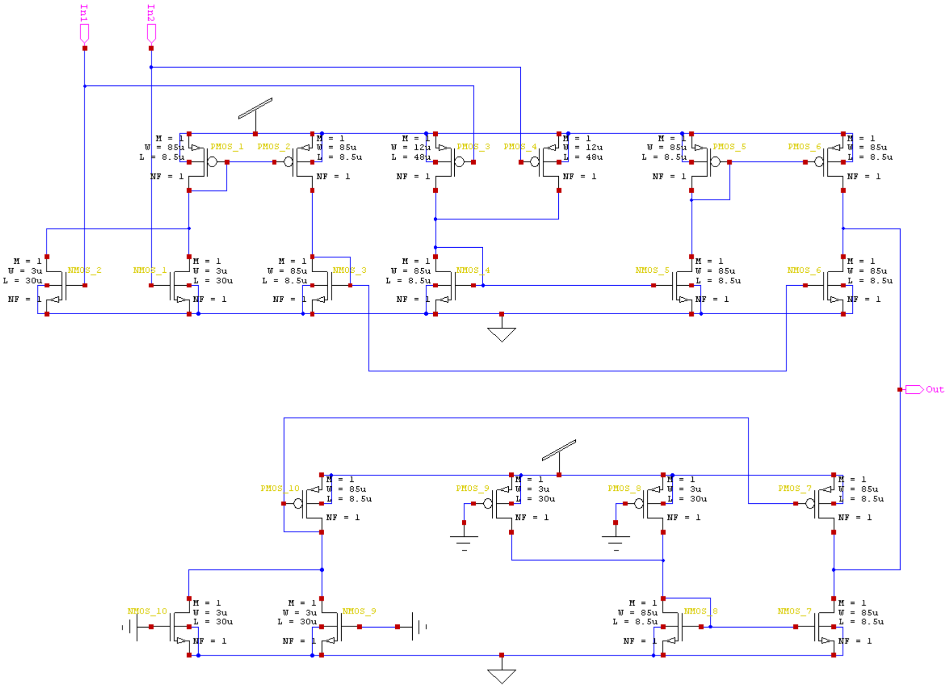

In1
In2
Out
PMOS_1 M = 1 W = 85u L = 8.5u NF = 1
PMOS_2 M = 1 W = 85u L = 8.5u NF = 1
PMOS_3 M = 1 W = 12u L = 48u NF = 1
PMOS_4 M = 1 W = 12u L = 48u NF = 1
PMOS_5 M = 1 W = 85u L = 8.5u NF = 1
PMOS_6 M = 1 W = 85u L = 8.5u NF = 1
NMOS_2 M = 1 W = 3u L = 30u NF = 1
NMOS_1 M = 1 W = 3u L = 30u NF = 1
NMOS_3 M = 1 W = 85u L = 8.5u NF = 1
NMOS_4 M = 1 W = 85u L = 8.5u NF = 1
NMOS_5 M = 1 W = 85u L = 8.5u NF = 1
NMOS_6 M = 1 W = 85u L = 8.5u NF = 1
PMOS_10 M = 1 W = 85u L = 8.5u NF = 1
PMOS_9 M = 1 W = 3u L = 30u NF = 1
PMOS_8 M = 1 W = 3u L = 30u NF = 1
PMOS_7 M = 1 W = 85u L = 8.5u NF = 1
NMOS_10 M = 1 W = 3u L = 30u NF = 1
NMOS_9 M = 1 W = 3u L = 30u NF = 1
NMOS_8 M = 1 W = 85u L = 8.5u NF = 1
NMOS_7 M = 1 W = 85u L = 8.5u NF = 1

Fig. 4.12. Schematics for a CMOS analog Inverter-Adder circuit (S-Edit implementation)

The CMOS implementation of the adder circuit with inverted output type is presented in Fig.4.12. In a similar way like for the adder circuit it can be shown that the output current for the Inverter/Adder is:

$$I_{out} = 1/2(\beta_p - \beta_n)\left(V_1^2 + V_2^2\right) - (C_n\beta_n + C_p\beta_p)(V_1 + V_2) \quad (4.25)$$

If

$$\beta_n \approx \beta_p \quad (4.26)$$

Then

$$V_{out} = -K(V_1 + V_2) \quad (4.27)$$

With

$$K = R_{out}(C_n\beta_n + C_p\beta_p) \quad (4.28)$$

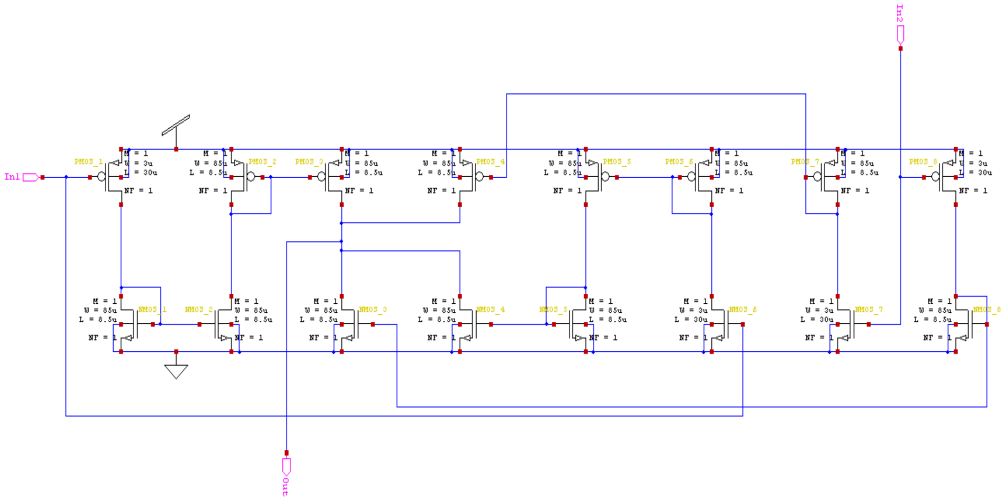


Fig. 4.13. Schematic for a CMOS analog Subtractor circuit (S-Edit implementation).

For the CMOS analog Subtractor circuit it can be proved that the output current is:

$$I_{out} = 1/2(\beta_n - \beta_p)\left(V_1^2 + V_2^2\right) + (C_n\beta_n + C_p\beta_p)(V_2 - V_1) \quad (4.29)$$

and if it is considered that:

$$\beta_n \approx \beta_p \quad (4.30)$$

then

$$V_{out} = K(V_2 - V_1) \quad (4.31)$$

with

$$K = R_{out}\left(C_n\beta_n + C_p\beta_p\right) \quad (4.32)$$

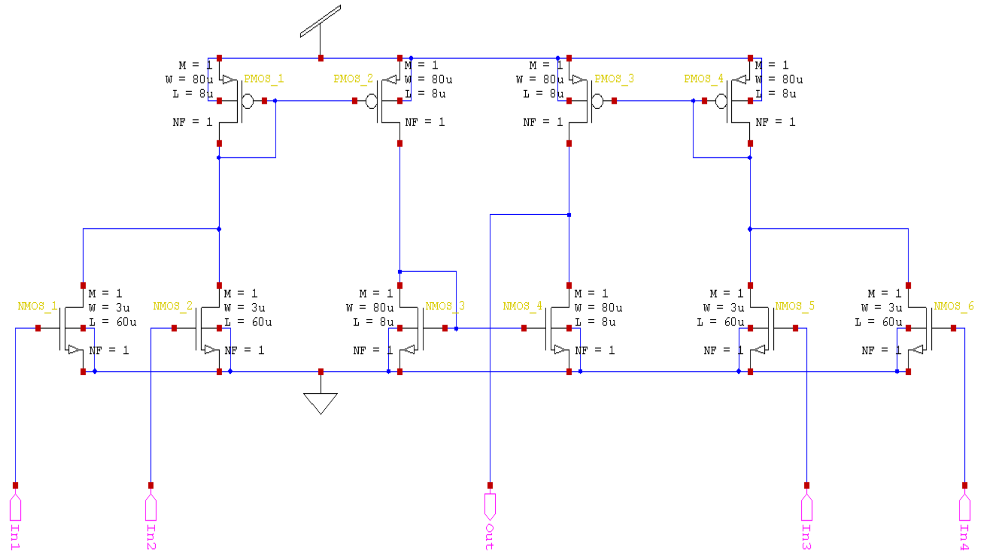


Fig. 4.14. Schematic for a four quadrant CMOS analog Multiplier using sub-circuits (S-Edit implementation).

The inputs for the CMOS four quadrant multiplier are the previous circuits: adder, inverter/adder and substractor. The output current is :

$$I_{out} = \beta_n\left(V_1^2 + 2V_1V_2 + V_1^2 + C_n^2\right) - \beta_2\left(V_1^2 + 2V_1V_2 + V_2^2 + C_n^2\right) = 4\beta_n V_1 V_2 \quad (4.33)$$

and

$$V_{out} = KV_1V_2 \quad (4.34)$$

With:

$$K = 4\beta_n R_{out} \quad (4.35)$$

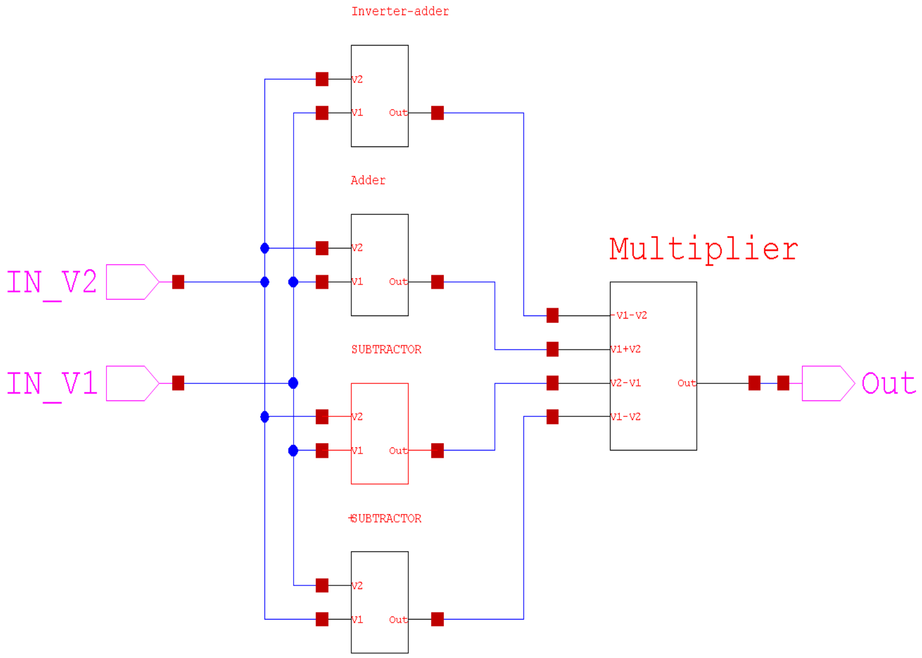


Fig. 4.15. Schematic for the analog CMOS multiplier using sub-circuits (S-Edit implementation).

4.1.5 Bio-inspired obstacle detection system using voltage-mode computational circuits

Based on the arithmetic CMOS circuit presented in 4.1.4 is proposed a bio-inspired obstacle detection system. In Fig. 4.16 is presented a one cell element from the entire bio-inspired obstacle detection system composed from the adaptive photoreceptor and the computational circuits. First, each signal received from the photoreceptor is computed like in Fig.4.16 using the structure of the elementary motion detector (EMD):

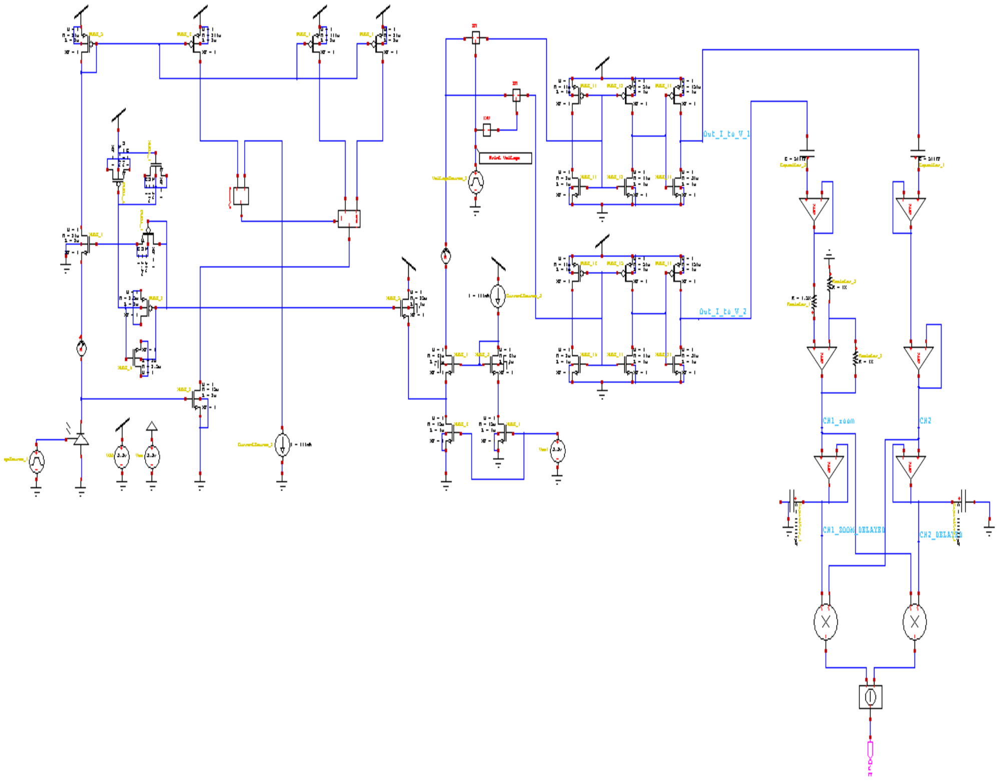


Fig. 4.16. One element of the proposed adaptive photoreceptor with voltage mode computational circuits (S-Edit implementation).

The DC component received from the photoreceptor is blocked since it does not contain any motion information and is distributed through a buffer at two channels. One channel amplifies the signal (approximately with a factor of 2) using a simple op-amp and resistors while the other channel leaves the signal unchanged with an op-amp in repeater configuration. The EMD structure is made using the previous CMOS arithmetic circuits (multiplier and subtractor). The delay from the EMD is achieved using a capacitor(~ 1 pF) and a Op-Amp in repeater configuration both forming a first-order low-pass filter with the time constant set to approximately 25 ms ued for biological motion detectors [105].

The one-cell element from Fig.4.16 was simulated supposing the same variation of the input current produced by the photodiode between 10nA and 100nA. The results of the simulation are presented in Fig.4.17 and Fig.4.18 showing the response of the element and also some intermediate signals. As can be seen when the photodiode changes the current generated from 10 nA to 100 nA, the output responds with a positive signal (Fig.4.18, DIFF) and when the photodiode changes the current from 100 nA to 10 nA responds with a negative signal. This is an ideal case because the 1-to-2 demultiplexer has the same period with the signal produced by the photodiode, but in reality this almost never happens.

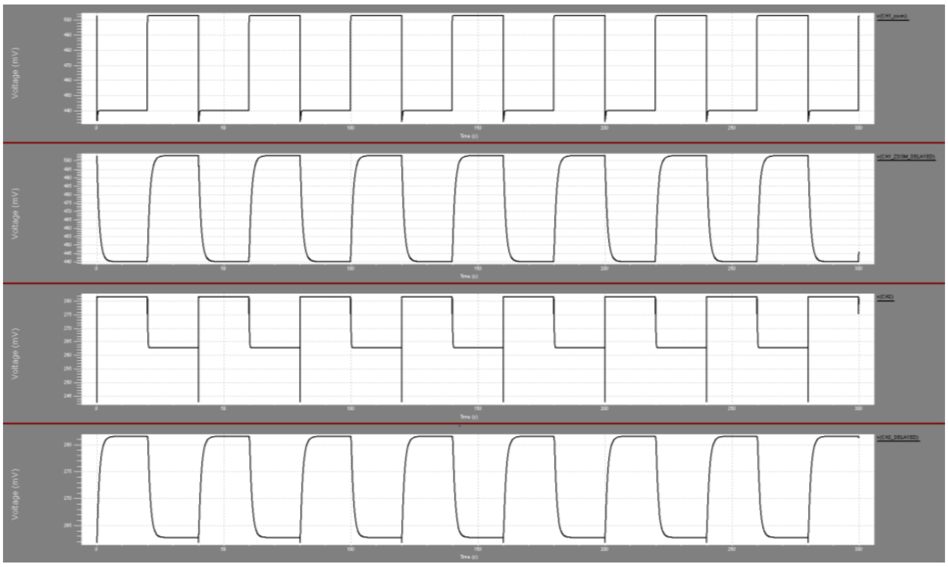


Fig. 4.17. Simulation of the schematic from Fig.4.16. Intermediate signals: V(CH1_ZOOM), V(CH1_ZOOM_DELAYED), V(CH2_nonZOOM), V(CH2_nonZOOM_DELAYED)

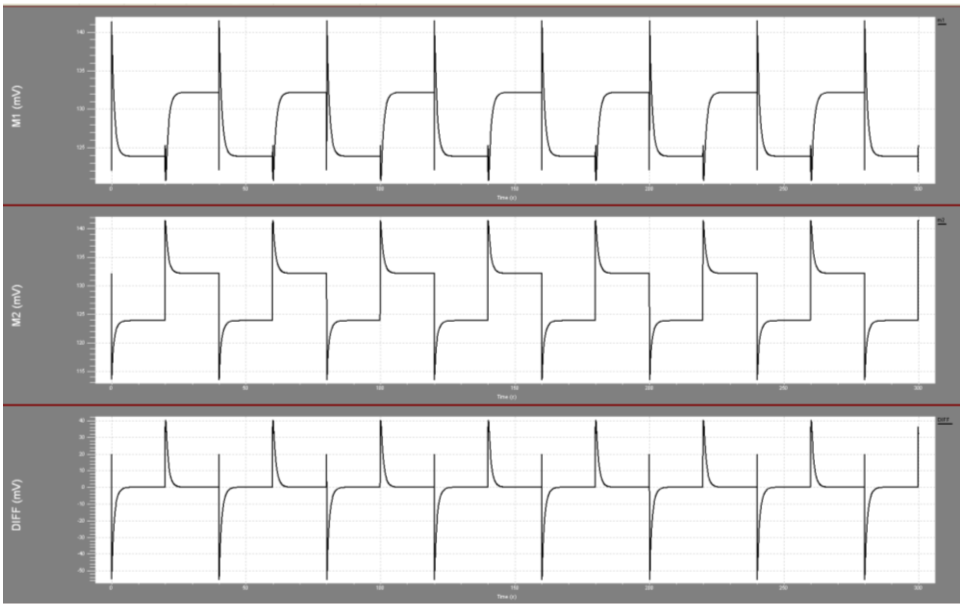


Fig. 4.18. Simulation of the schematic from Fig.4.16. Intermediate signals: M1 – result of the multiplication between V(CH1_ZOOM_DELAYED) and V(CH2_nonZOOM); M2 - result of the multiplication between V(CH1_ZOOM) and V(CH2_nonZOOM_DELAYED); DIFF – result of the subtraction between M1 and M2

Let’s suppose that the 1-to-2 demultiplexer is commanded with a different frequency, (period) than has the photodiode (T_{PHD}= 40s; T_{DEMUX}=15s).

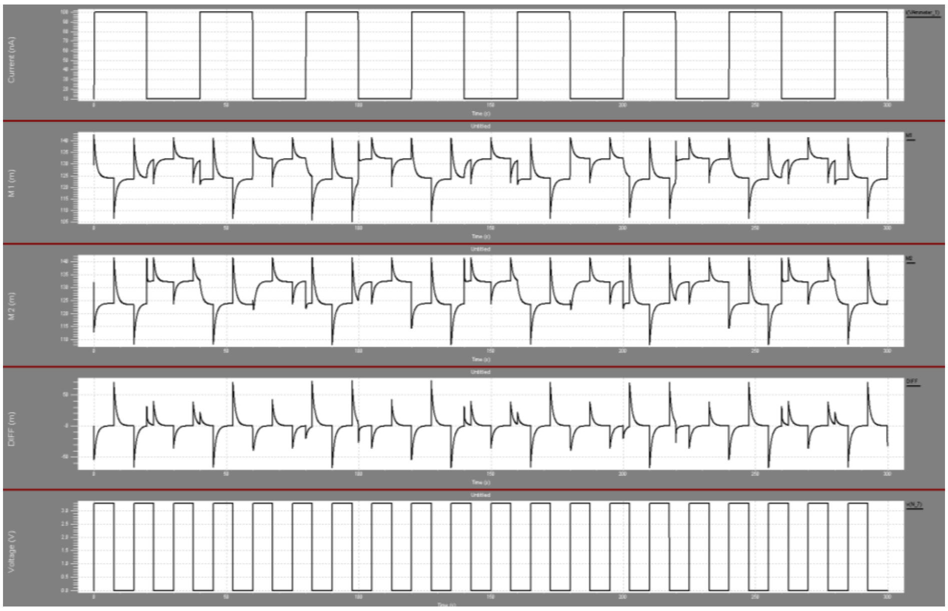


Fig. 4.19 Simulation of the schematic from Fig.4.16. Intermediate signals:I(PHD) – current generated by the photodiode; M1 – result of the multiplication between V(CH1_ZOOM_DELAYED) and V(CH2_nonZOOM); M2 - result of the multiplication between V(CH1_ZOOM) and V(CH2_nonZOOM_DELAYED); DIFF – result of the subtraction between M1 and M2; Vctrl – voltage control of the demultiplexer

In Fig. 4.19 it can be seen that the output (DIFF) generates false responses in almost all the situations, so only when the control of the demultiplexer is sincronized with the changes of the current generated by the photodiode the output is correct. This issue brought the idea of a new proposed circuit where this feature is included. In the new circuit (Fig. 4.20) the demultiplexer is controlled by the output of the current comparator, so in this way the changes in the input current is synchronized with the channels control.

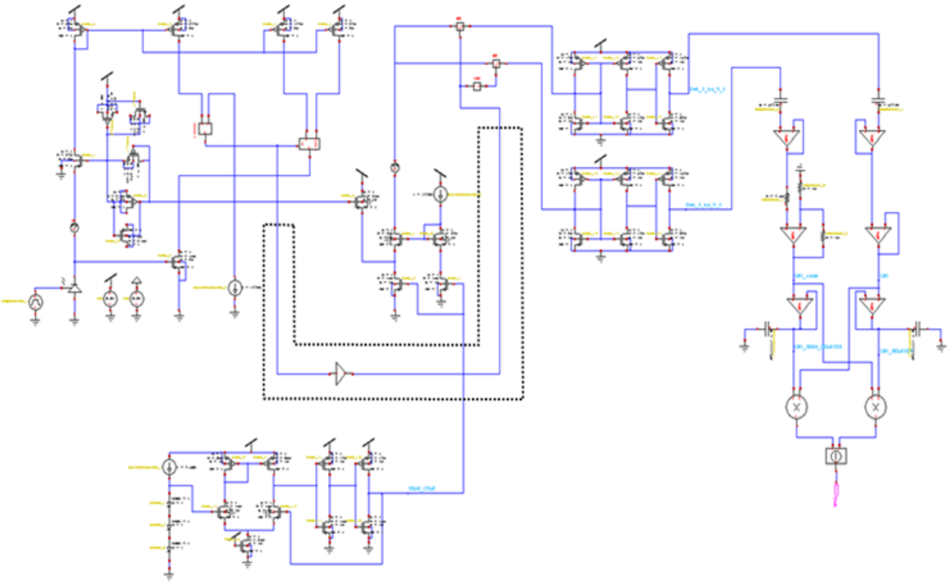


Fig. 4.20. One element of the proposed adaptive photoreceptor with voltage mode computational circuits and synchronization feature (S-Edit implementation).

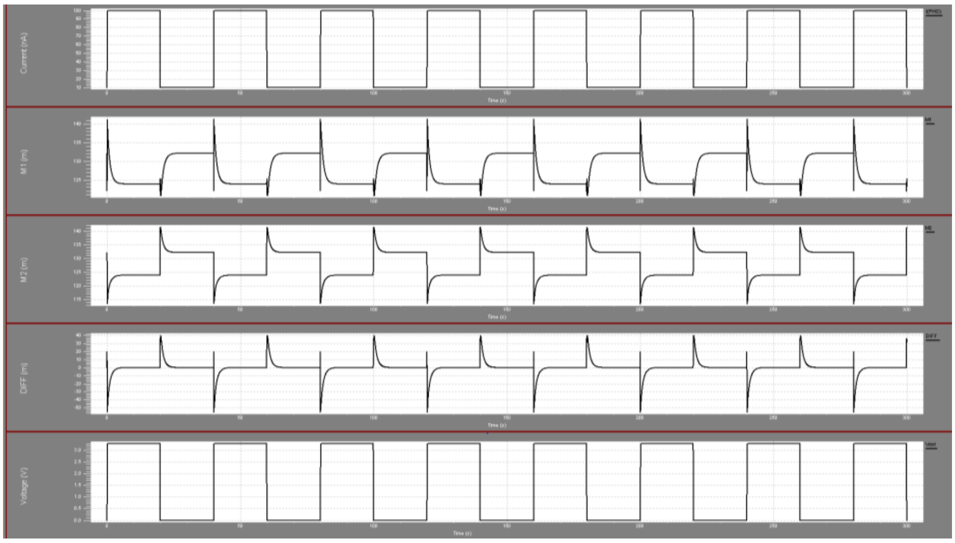


Fig. 4.21 Simulation of the schematic from Fig.4.20. Intermediate signals:I(PHD) – current generated by the photodiode; M1 – result of the multiplication between V(CH1_ZOOM_DELAYED) and V(CH2_nonZOOM); M2 - result of the multiplication between V(CH1_ZOOM) and V(CH2_nonZOOM_DELAYED); DIFF – result of the subtraction between M1 and M2; Vctrl – voltage control of the demultiplexer

Combining the proposed adaptive-photoreceptor and computational elements presented earlier and following the algorithm of the proposed fly-locust bio-inspired obstacle detection system presented in Chapter 3 (block diagram presented in Fig. 4.22), a structure of a 25 resolution obstacle detection system in CMOS technology can be implemented like in Fig.4.19. The adder and the inverter-adder from Fig.4.23 were implemented using the classic operational amplifier based computation circuits.

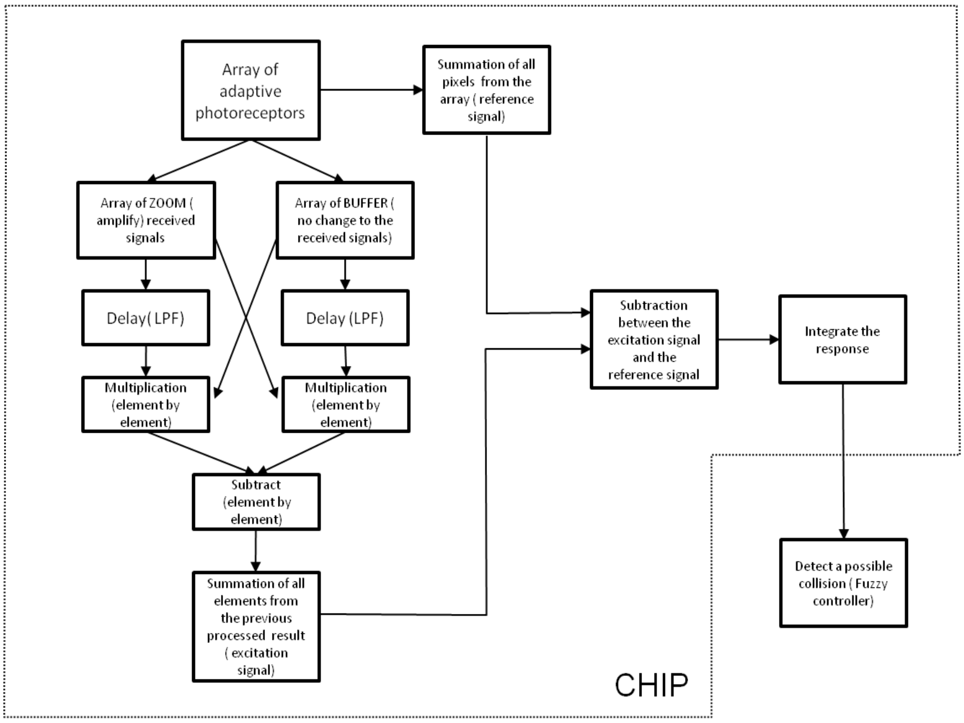

Array of adaptive photoreceptors
Summation of all pixels from the array (reference signal)
Array of ZOOM (amplify) received signals
Array of BUFFER (no change to the received signals)
Delay(LPF)
Delay (LPF)
Multiplication (element by element)
Multiplication (element by element)
Subtract (element by element)
Summation of all elements from the previous processed result (excitation signal)
Subtraction between the excitation signal and the reference signal
Integrate the response
Detect a possible collision (Fuzzy controller)
CHIP

Fig. 4.22. Block diagram concept for the bio-inspired obstacle detection system

The 25 resolution was choose arbitrary for presentation propose only, the resolution for a real implementation has to be choose depending on how much permits the technology (power consumption, costs).

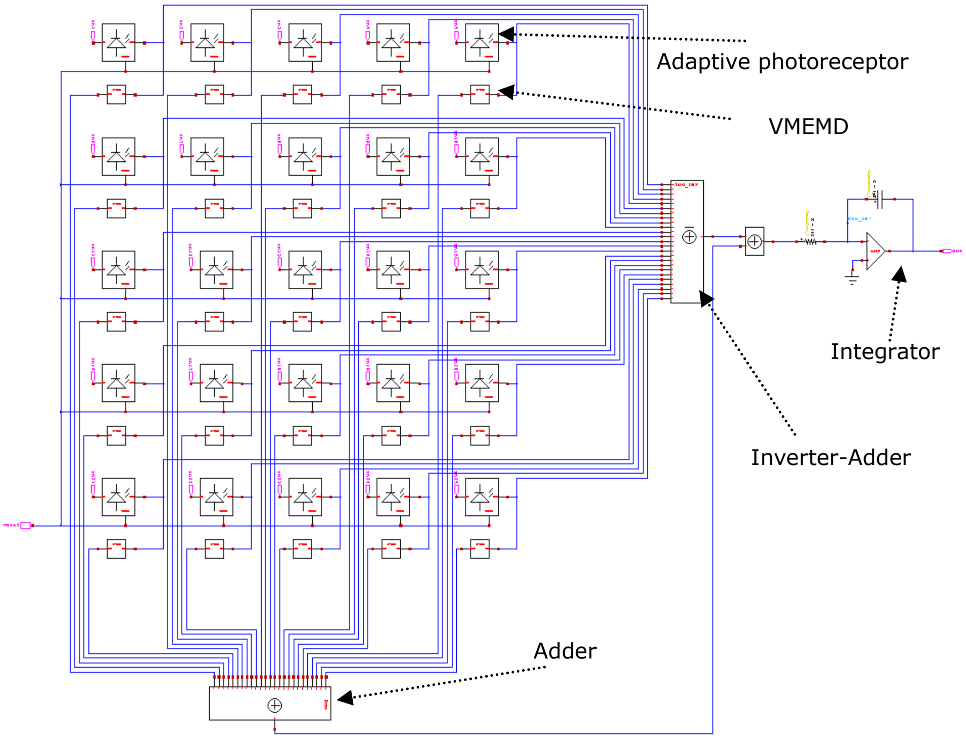


Fig. 4.23. 25 resolution CMOS chip concept for the bio-inspired obstacle detection system (S-edit implementation)

4.1.6 Temperature influence on the voltage mode bio-inspired obstacle detection system functionality

One important issue regarding the correct functionality of this kind of system is the behavior with the temperature variation. If the response of the adaptive photoreceptor is not correct then the error will be propagated to the computational circuits and in spite of a good functionality of these circuits, the output will be with high deviations. Next will be analyzed the variation with the temperature and will be proposed an additional circuit for temperature compensation.

In Fig.4.22 is shown the current variation with temperature for the output current of the half-gaussian circuit when the photodiode produce a variation between 10 nA and 100 nA in the case without temperature compensation. As can be seen when the temperature is modified between -15°C and +75°C the output current vary with about 50 nA which is not negligible. Using an additional temperature compensation circuit the variation is considerable reduced (see Fig.4.23) nearly to 12 nA ,so 76 % variation reduction from the initial case. Also the variation with temperature of the intermediate voltages across the computational part is significantly reduced (Fig.4.24, Fig.4.25) in particular at low temperatures.

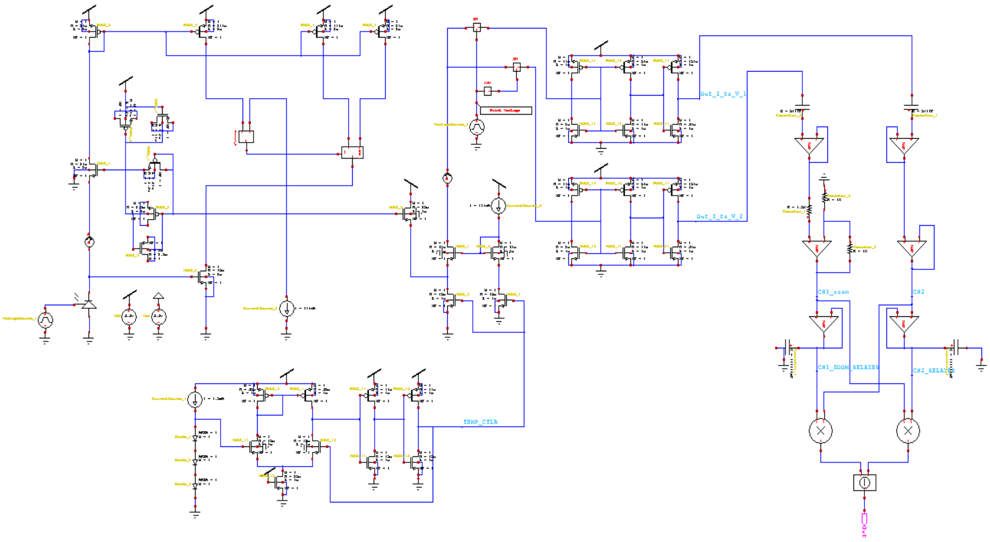


Fig. 4.24. One element of the adaptive photoreceptor with the voltage mode computational circuits and temperature compensation circuit.

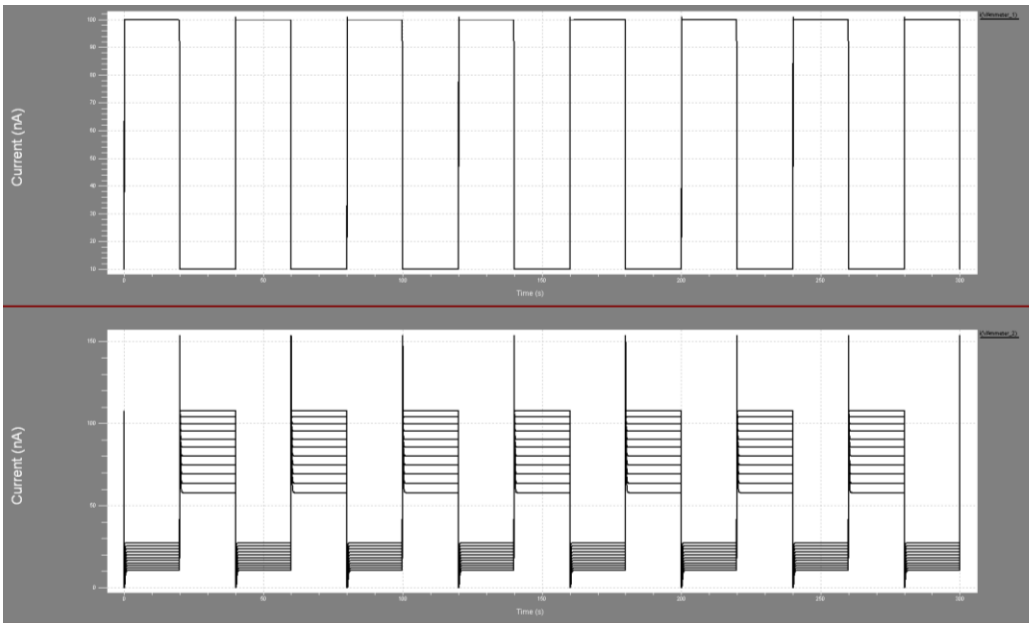


Fig. 4.25. Simulation of the schematic from Fig.4.21. in the case **without** temperature compensation for a temperature variation between -15°C and 75°C . Current generated by the photodiode. Response of the Half-Gaussian circuit.

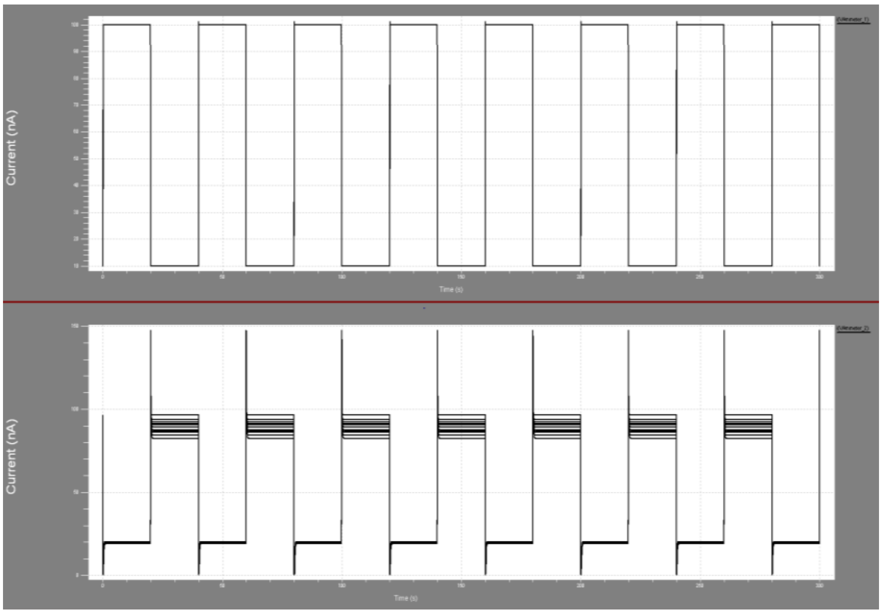


Fig. 4.26. Simulation of the schematic from Fig.4.21. in the case **with** temperature compensation for a temperature variation between -15°C and 75°C . Current generated by the photodiode. Response of the Half-Gaussian circuit.

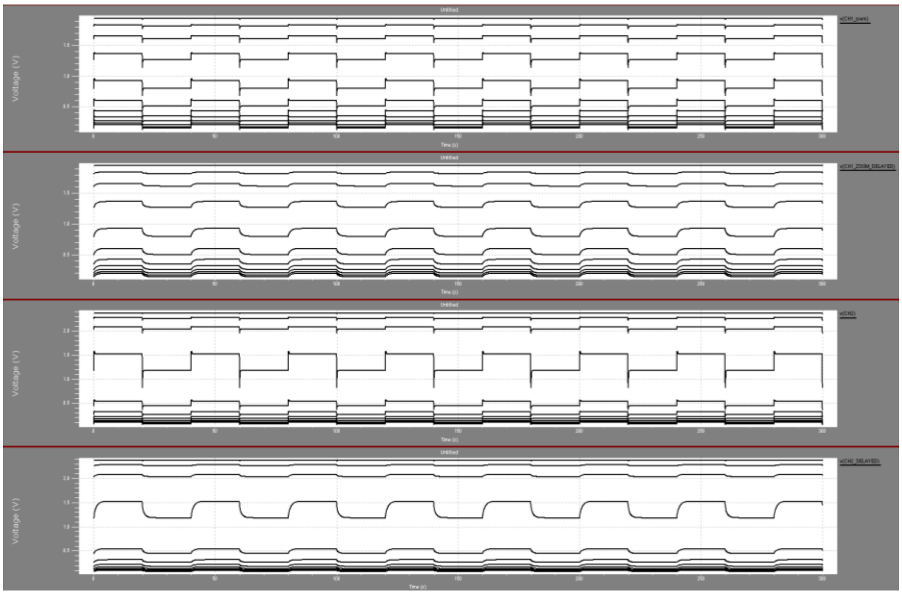


Fig. 4.27. Simulation of the schematic from Fig.4.21. in the case **without** temperature compensation for a temperature variation between -15°C and 75°C . Intermediate signals: V(CH1_ZOOM), V(CH1_ZOOM_DELAYED), V(CH2_nonZOOM), V(CH2_nonZOOM_DELAYED)

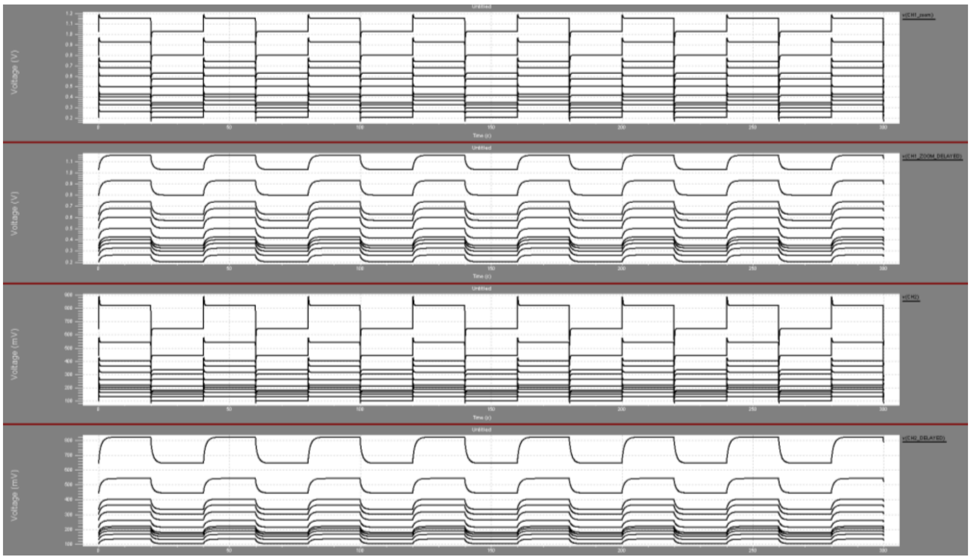


Fig. 4.28. Simulation of the schematic from Fig.4.21. in the case **with** temperature compensation for a temperature variation between -15°C and 75°C . Intermediate signals: V(CH1_ZOOM), V(CH1_ZOOM_DELAYED), V(CH2_nonZOOM), V(CH2_nonZOOM_DELAYED)

The temperature compensation circuit described in [106] is using 3 diodes in a series connection biased from a constant current source and a repeater amplifier. Since the forward voltage on a diode is temperature dependent it will be used to control the half-gaussian circuit response. Connecting the control voltage from the temperature compensation circuit to the half-gaussian circuit has also the advantage of eliminating an additional bias circuit.

4.1.7 Bio-inspired obstacle detection system using current-mode computational circuits

This chapter proposes an approach for the bio-inspired obstacle detection system using current mode CMOS computational circuits for the visual processing part. In order to implement in this mode, for the delay part was used Current Mode Low Pass Filter (CMLPF) [101] (see Fig.4.26) and for the computation part Current Mode Multipliers [99], [101] (see Fig.4.28, Fig.4.29 and Fig.4.30) and Current Mode Subtractor [100] (see Fig.4.30), and all of this circuits will be described next.

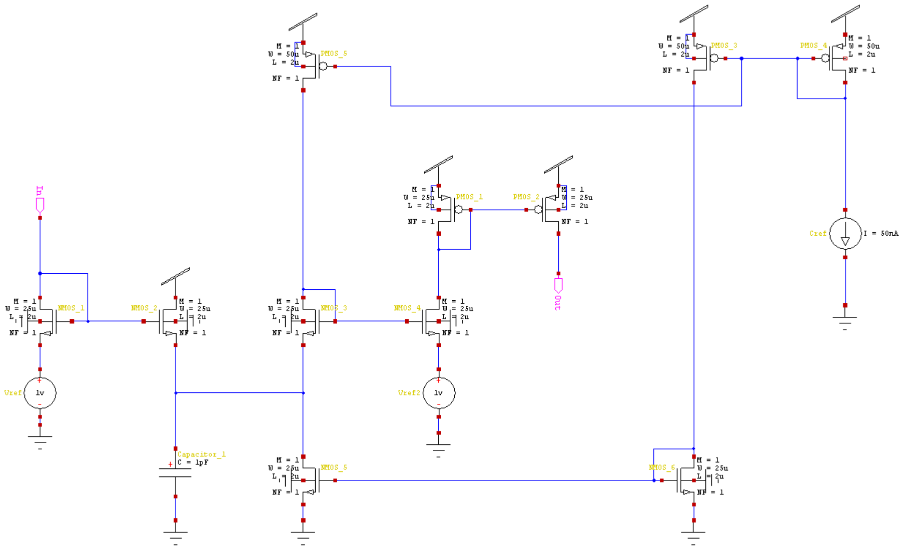


Fig. 4.29. Current mode low pass filter (CMLPF) (S-edit implementation)

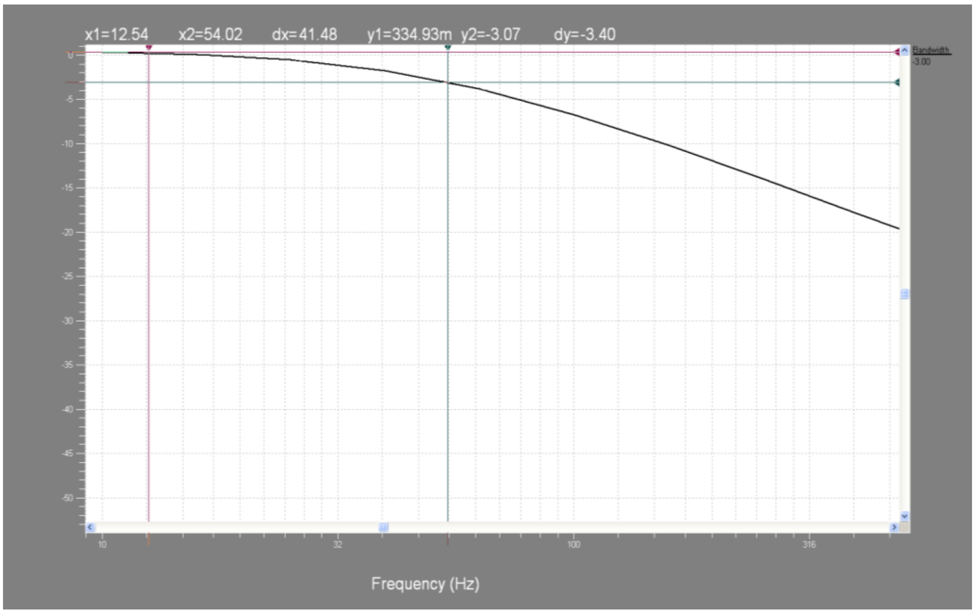


Fig. 4.30. Response of the CMLPF.

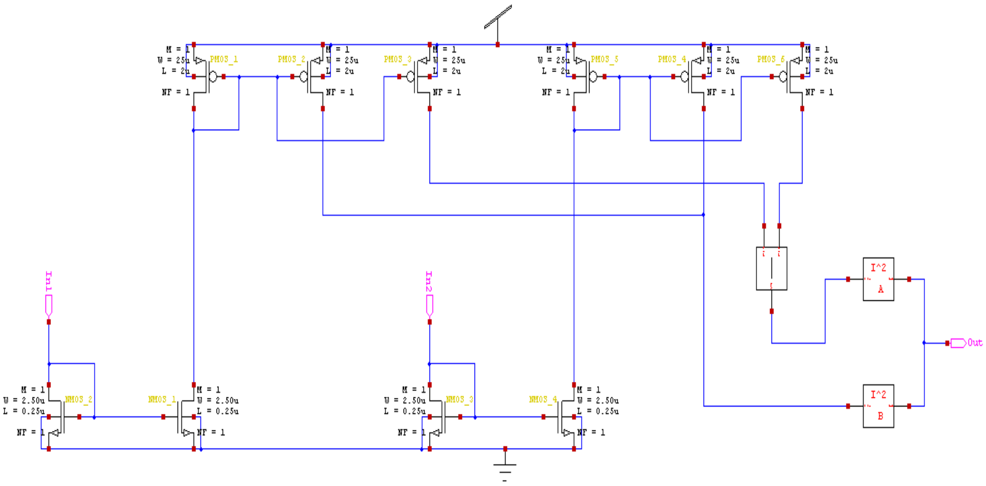


Fig. 4.31. Current mode multiplier (S-edit implementation)

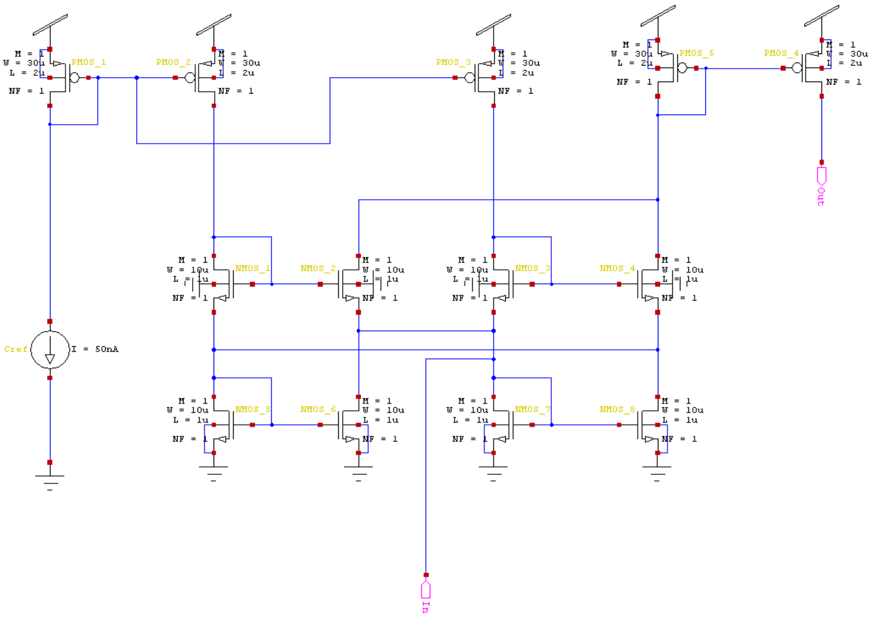


Fig. 4.32. Current squaring circuit with PMOS mirror output type (S-edit implementation)

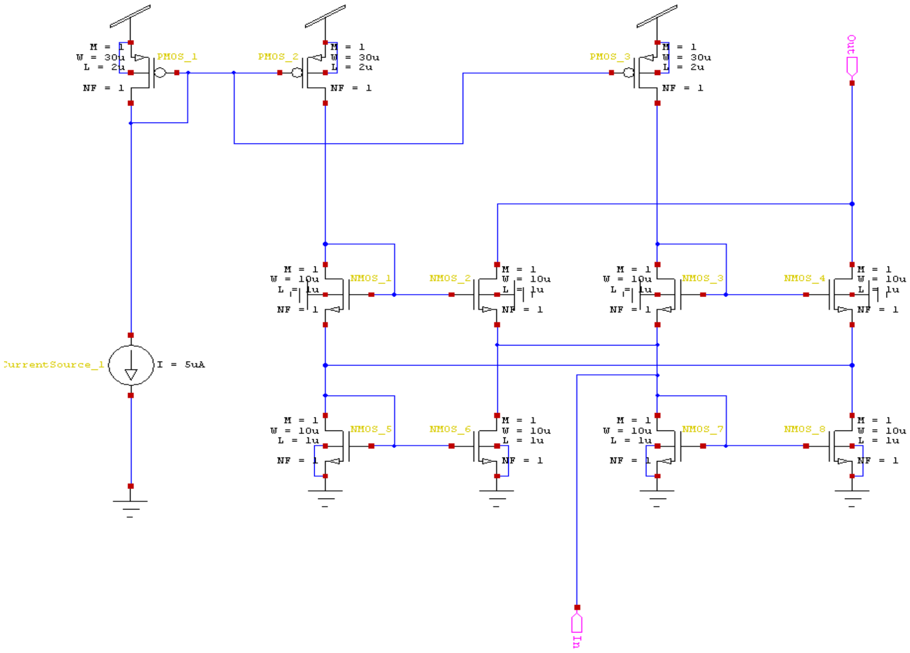


Fig. 4.33. Current squaring circuit with NMOS mirror output type (S-edit implementation)

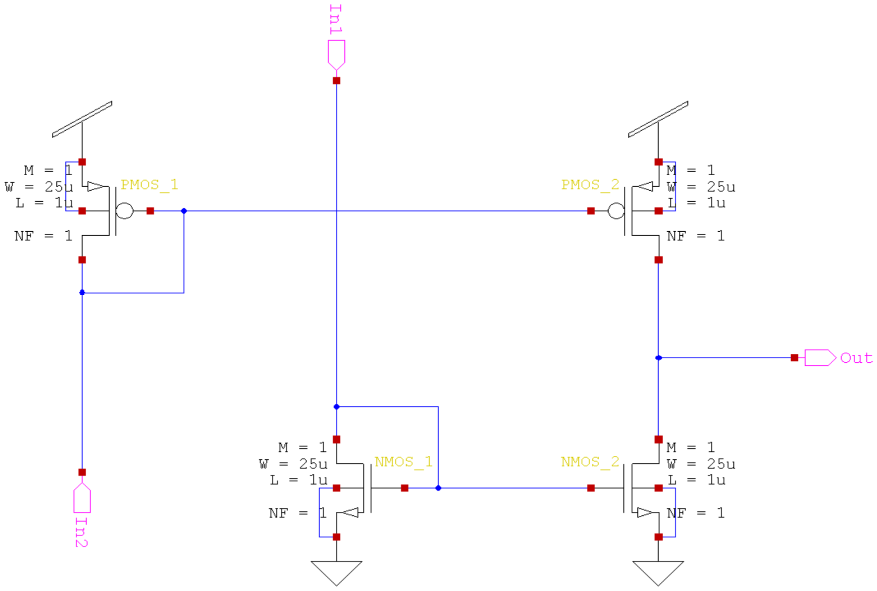


Fig. 4.34. Simple current subtractor (S-edit implementation).

Based on [99] it was used a CMOS current multiplier. The circuit uses the inherited square-law characteristics of the MOS transistors and is mainly constructed from squaring circuits, current mirrors and a current subtractor.

The drain current of a transistor in saturation region can be written as:

$$I_d = k(V_{gs} - V_t)^2 \quad (4.36)$$

With:

$$k = \frac{1}{2}\mu C_{ox}\frac{W}{L} \quad (4.37)$$

So,

$$V_{gs} = V_t + \sqrt{I_d / k} \quad (4.38)$$

Using the Kirchhoff's voltage laws for the group of the MOS transistors connected in a translinear loop result that:

$$V_{gs1} + V_{gs2} = V_{gs3} + V_{gs4} \quad (4.39)$$

$$V_{t1} + \sqrt{\frac{I_1}{k_1}} + V_{t2} + \sqrt{\frac{I_2}{k_2}} = V_{t3} + \sqrt{\frac{I_3}{k_3}} + V_{t4} + \sqrt{\frac{I_4}{k_4}} \quad (4.40)$$

The threshold voltages for the transistor M1 and M4 are equal , and also for the transistor M2 and M3, so :

$$\sqrt{I_1} + \sqrt{I_2} = \sqrt{I_3} + \sqrt{I_4} \quad (4.41)$$

The currents I_1 and I_2 are each forced equal with the reference current, so :

$$\sqrt{I_3} + \sqrt{I_4} = \sqrt{4I_B} \quad (4.42)$$

$$I_3 + I_4 + 2\sqrt{I_3 I_4} = 4I_B \quad (4.43)$$

$$(I_3 + I_4 - 4I_B)^2 = 4I_3 I_4 \quad (4.44)$$

$$I_3^2 + 2I_3 I_4 + I_4^2 - 8I_B(I_3 + I_4) + 16I_B^2 = 4I_3 I_4 \quad (4.45)$$

$$(I_3 - I_4)^2 + 16I_B^2 = 8I_B(I_3 + I_4) \quad (4.46)$$

$$I_3 + I_4 = 2I_B + \frac{(I_3 - I_4)^2}{8I_B} = I_{out} \quad (4.47)$$

From the current mirror results that:

$$I_{in} + I_2 + I_3 = I_1 + I_4 \quad (4.48)$$

$$I_{in} = I_4 - I_3 \quad (4.49)$$

So the output current of the squaring circuit will be :

$$I_{out} = 2I_B + \frac{I_{in}^2}{8I_B} \quad (4.50)$$

It is important to mention that this relation is valid only if the transistor work in the saturation region. According to [99] current I_3 and I_4 have to be greater or equal to zero, so:

$$I_3 = I_B - \frac{I_{in}}{2} + \frac{I_{in}^2}{16I_B} \quad (4.51)$$

$$I_4 = I_B + \frac{I_{in}}{2} + \frac{I_{in}^2}{16I_B} \quad (4.52)$$

with:

$$-4I_B \le I_{in} \le 4I_B \quad (4.53)$$

Using two squaring circuits the output of the current multiplier in the configuration from Fig.4.28 , can be expressed as:

$$I_{out} = I_{o1} - I_{o2} \quad (4.54)$$

Knowing that

$$I_{o1} = 2I_B + \frac{(I_y + I_x)^2}{8I_B} \quad (4.55)$$

$$I_{o2} = 2I_B + \frac{(I_y - I_x)^2}{8I_B} \quad (4.56)$$

the output current of the multiplier becomes :

$$I_{out} = I_{o1} - I_{o2} = \frac{I_x I_y}{2I_B} \quad (4.57)$$

with the condition:

$$\left|I_x\right| + \left|I_y\right| \leq 4I_B \quad (4.58)$$

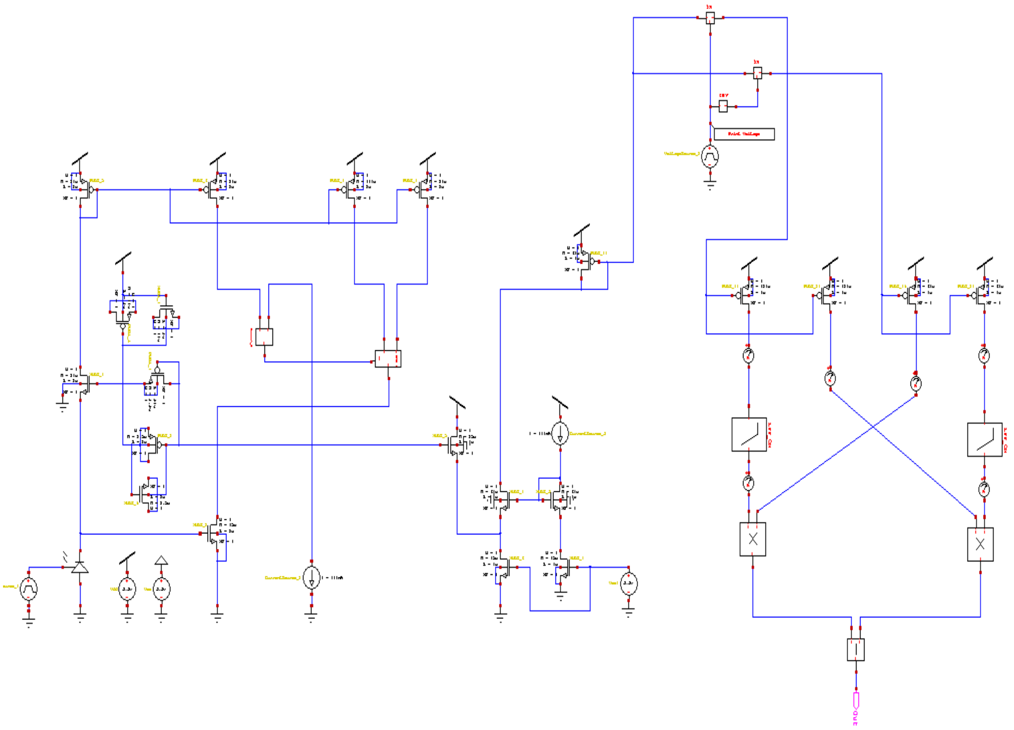


Fig. 4.35. One element of the proposed adaptive photoreceptor with current mode computational circuits (S-edit implementation).

The schematic from Fig.4.35 shows the implementation of one logarithmic photo-adaptive element with the computational part in current mode. Since the computational part is made in current mode, the current-to-voltage converter used before is not necessary anymore, the output current form the half-gaussian circuit is transferred to the computational circuits through a PMOS current mirror.

For the simulation was used the same case with current variation produced by the photodiode between 10 nA and 100 nA and the results of the simulation containing the variation of the signals in different points are shown in Fig.4.36 and Fig.4.37.

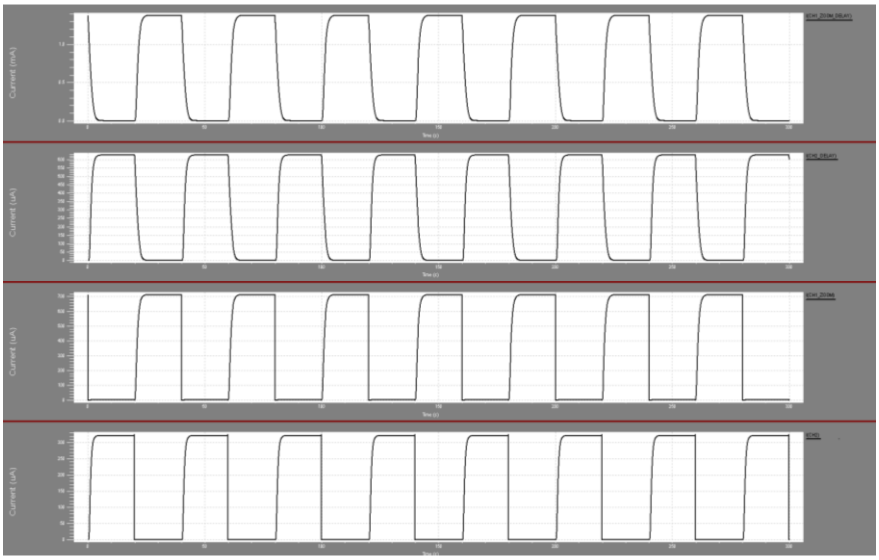


Fig. 4.36. Simulation of the schematic from Fig.4.35. Intermediate signals: I(CH1_ZOOM), I(CH1_ZOOM_DELAYED), I(CH2_nonZOOM), I(CH2_nonZOOM_DELAYED)

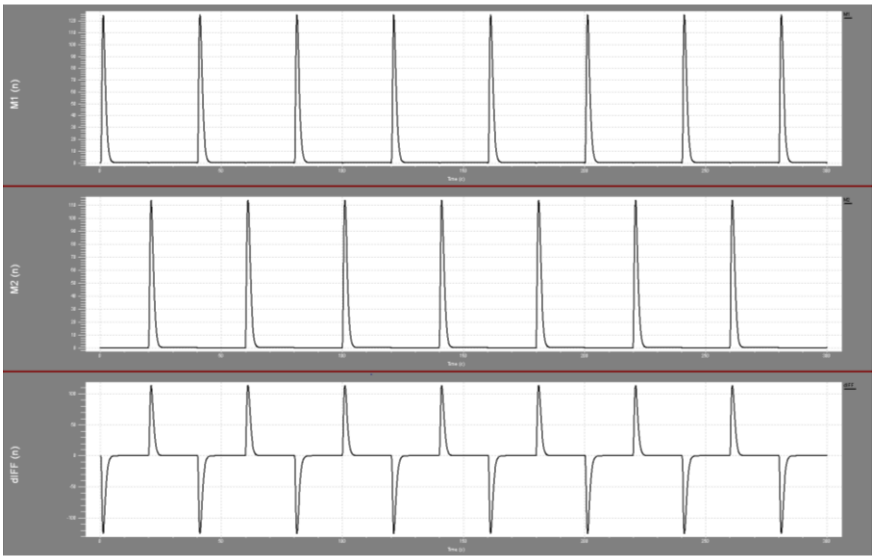


Fig. 4.37. Simulation of the schematic from Fig.4.35. Intermediate signals: M1 – result of the multiplication between I(CH1_ZOOM_DELYED) and I(CH2_nonZOOM); M2 - result of the multiplication between I(CH1_ZOOM) and I(CH2_nonZOOM_DELAYED); DIFF – result of the subtraction between M1 and M2

The same situation like in the case for the voltage computational mode happens here too (Fig.4.38). If the period of the demultiplexer control signal is different than the period of the photodiode current, the output generates false responses , so using the output of the current comparator to control the demultiplexer solves this issue (Fig.4.39, Fig.4.40).

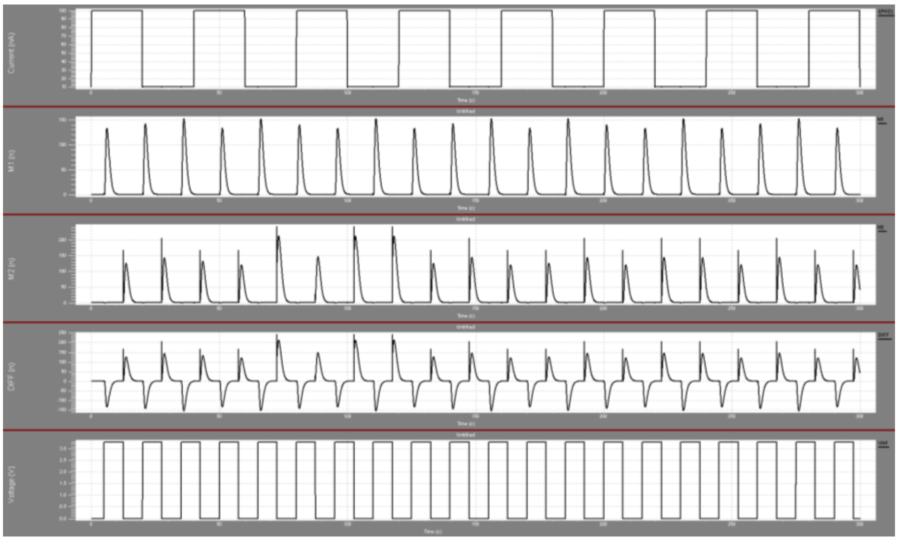


Fig. 4.38. Simulation of the schematic from Fig.4.35. Intermediate signals: I(PHD) – current generated by the photodiode; M1 – result of the multiplication between I(CH1_ZOOM_DELYED) and I(CH2_nonZOOM); M2 - result of the multiplication between I(CH1_ZOOM) and I(CH2_nonZOOM_DELAYED); DIFF – result of the subtraction between M1 and M2Intermediate signals: Vctrl – voltage control of the demultiplexer

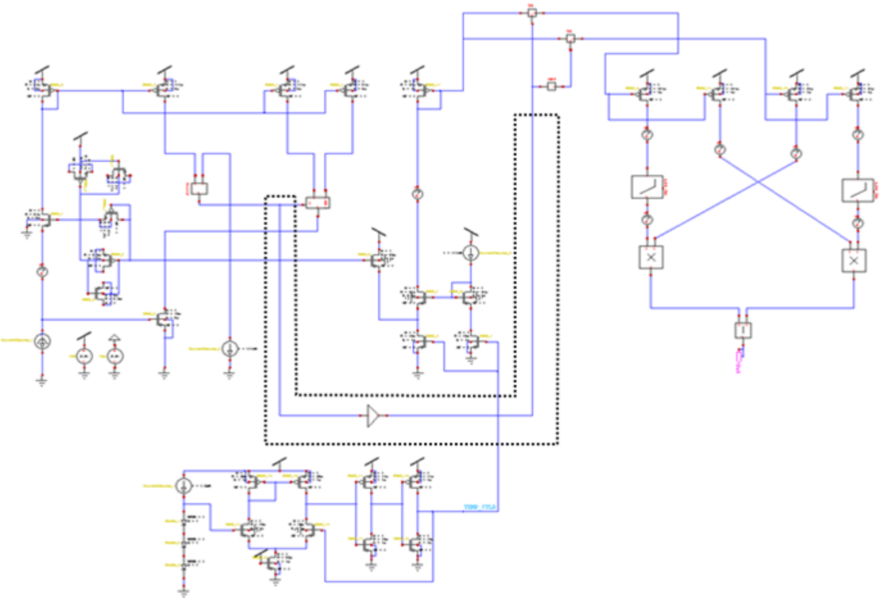


Fig. 4.39 One element of the proposed adaptive photoreceptor with current mode computational circuits (S-edit implementation).

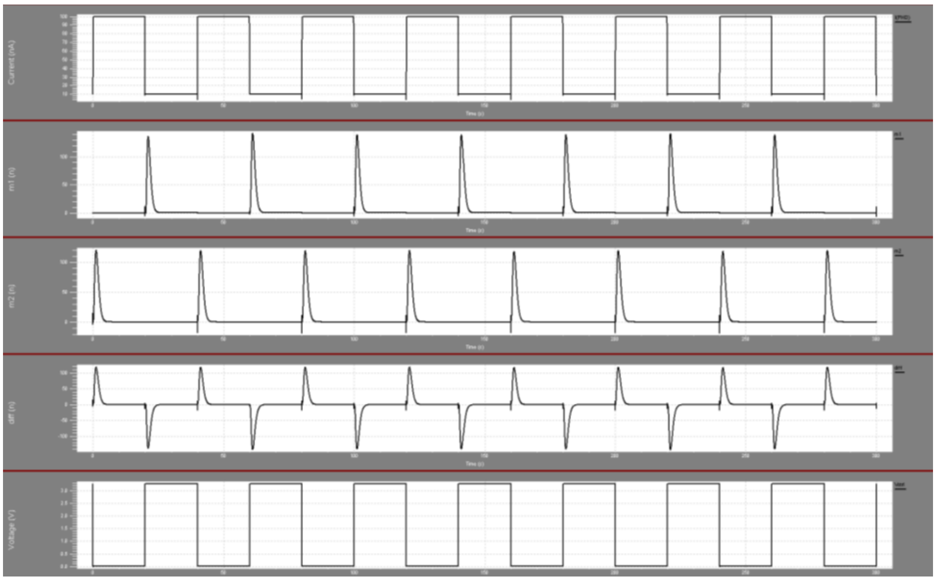


Fig. 4.40 Simulation of the schematic from Fig.4.35. Intermediate signals: I(PHD) – current generated by the photodiode; M1 – result of the multiplication between I(CH1_ZOOM_DELYED) and I(CH2_nonZOOM); M2 - result of the multiplication between I(CH1_ZOOM) and I(CH2_nonZOOM_DELAYED); DIFF – result of the subtraction between M1 and M2Intermediate signals: Vctrl – voltage control of the demultiplexer

4.1.8 Temperature influence on the current mode bio-inspired obstacle detection system functionality

In the same manner like for the voltage mode bio-inspired obstacle detection system, the temperature variation influences the functionality of the system. Next will be evaluated this variation with temperature for the case with and without an additional temperature compensation circuit (Fig.4.41).

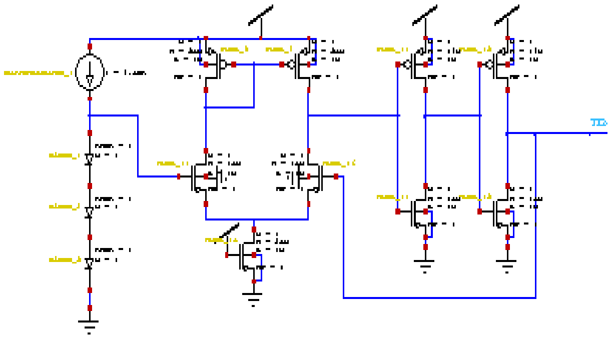


Fig. 4.41. Temperature compensation circuit (S-edit implementation)

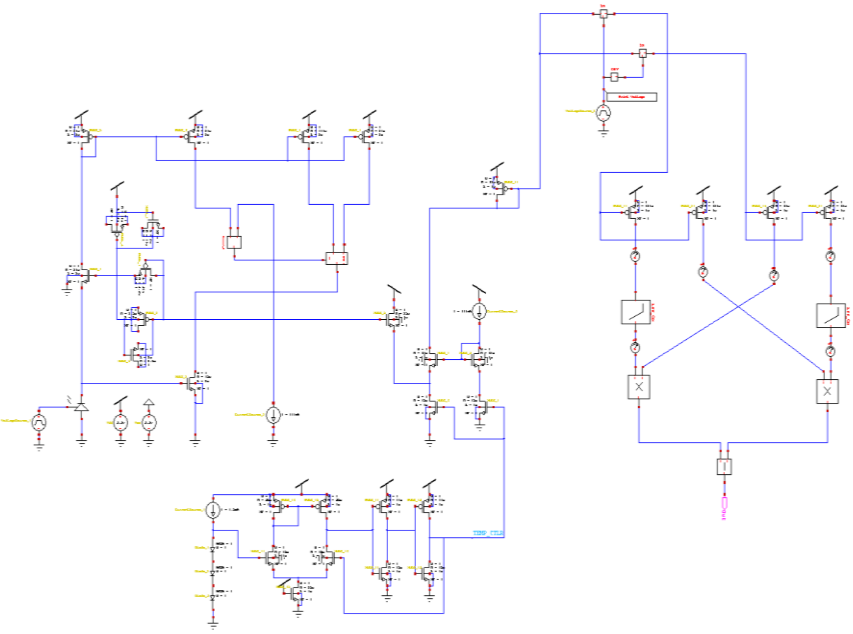


Fig. 4.42. One element of the proposed adaptive photoreceptor with current mode computational circuits and temperature compensation (S-edit implementation).

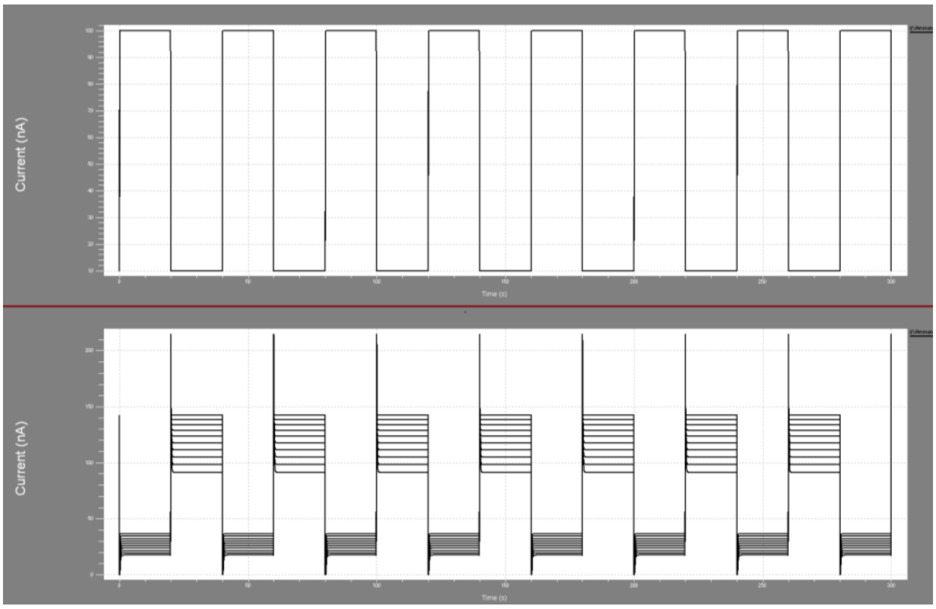


Fig. 4.43. Simulation of the schematic from Fig.4.35. in the case **without** temperature compensation for a temperature variation between -15°C and 75°C . Current generated by the photodiode. Response of the Half-Gaussian circuit.

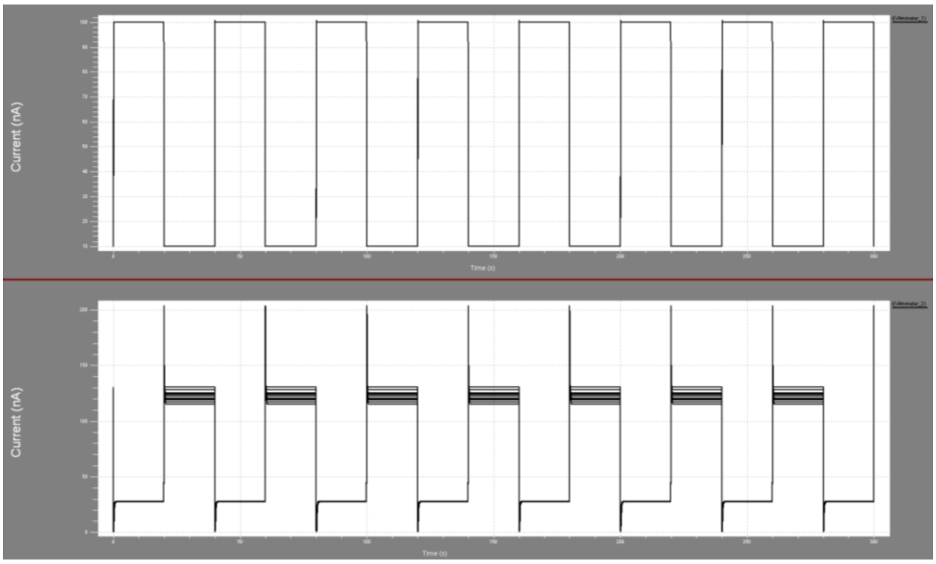


Fig. 4.44. Simulation of the schematic from Fig.4.35. in the case **with** temperature compensation for a temperature variation between -15°C and 75°C . Current generated by the photodiode. Response of the Half-Gaussian circuit.

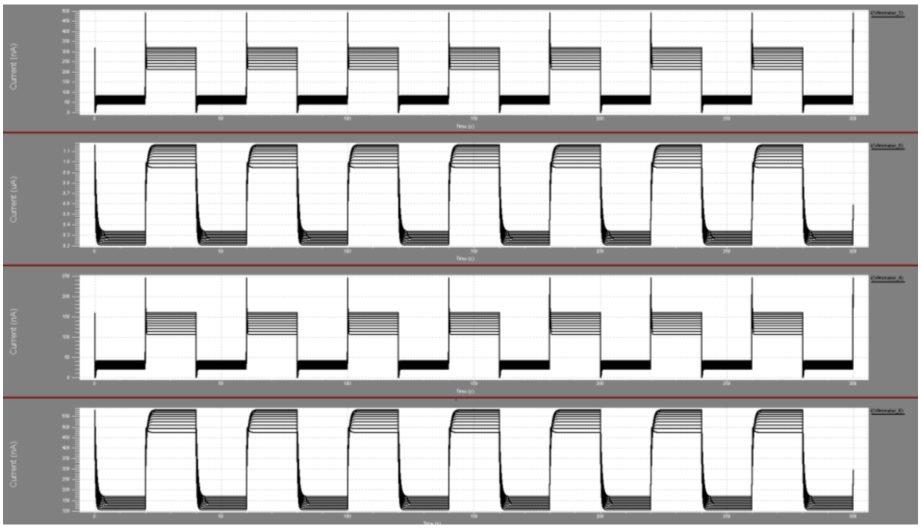


Fig. 4.45. Simulation of the schematic from Fig.4.35. in the case **without** temperature compensation for a temperature variation between -15°C and 75°C . Intermediate signals: I(CH1_ZOOM), I(CH1_ZOOM_DELAYED), I(CH2_nonZOOM), I(CH2_nonZOOM_DELAYED)

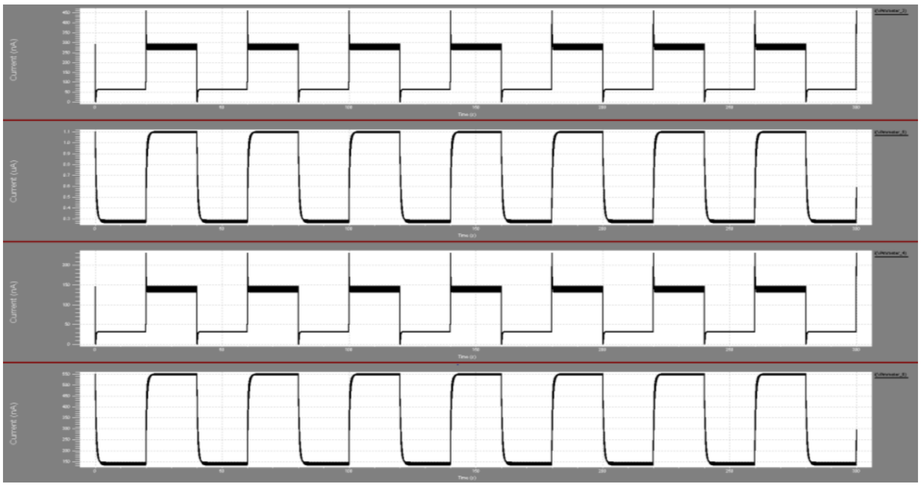


Fig. 4.46. Simulation of the schematic from Fig.4.35. in the case **with** temperature compensation for a temperature variation between -15°C and 75°C . Intermediate signals: I(CH1_ZOOM), I(CH1_ZOOM_DELAYED), I(CH2_nonZOOM), I(CH2_nonZOOM_DELAYED)

The simulation results from Fig.4.44 to Fig.4.46 shows a significant improvement when the temperature compensation circuit is used. In addition, the variation of the currents through the computational part is better than for the case of the voltage mode because for the current mode type, the current-to-voltage circuit is missing and all the processing is made in current.

4.2 Embedded solution using a smartphone

In our days is very hard to picture a world without smartphones. From the first smarthphone available on the market in 1993, the so called IBM Simon who had minor applications such as fax or pager, the smarthphones have evolved into an all-in-one device that can be use to surf the internet, to take high resolution pictures or record high definition videos, to listen music, watch TV, navigate using GPS, etc. One difference between these types of mobile phones is the operating system (OS) and there are a few one available like: Google's Android, Apple's iOS, Nokia's Symbian, RIM's BlackBerry OS, Samsung's Bada, Microsoft's Windows Phone, HP's webOS, and embedded Linux distributions such as Maemo and MeeGo [107].
From the point of view of the development engineer are very important the ones that are open source types giving the opportunity to the user to become also a developer and to create new applications.
A simplified version of the concept of the bio-inspired obstacle detection system presented in the Chapter 2 and 3, was implemented as an application for a smarthphone using Android operating system. Since a smarthphone contains a camera sensor, video processing capability and speaker for noticing signals it is a very good candidate to implement such an application. The advantages of using it are considerable: low cost (virtual zero costs if someone has already in possession a smarthphone), continuous improvement (also without additional costs), easy to use.

4.2.1 Description of the software based implementation method

Using java language in Eclipse tool with Android SDK plugin and OpenCV source library it was developed an obstacle detection application based on the bio-inspired algorithm presented in Chapter 2 and 3. The flow chart for the application is presented next in Fig.4.47:

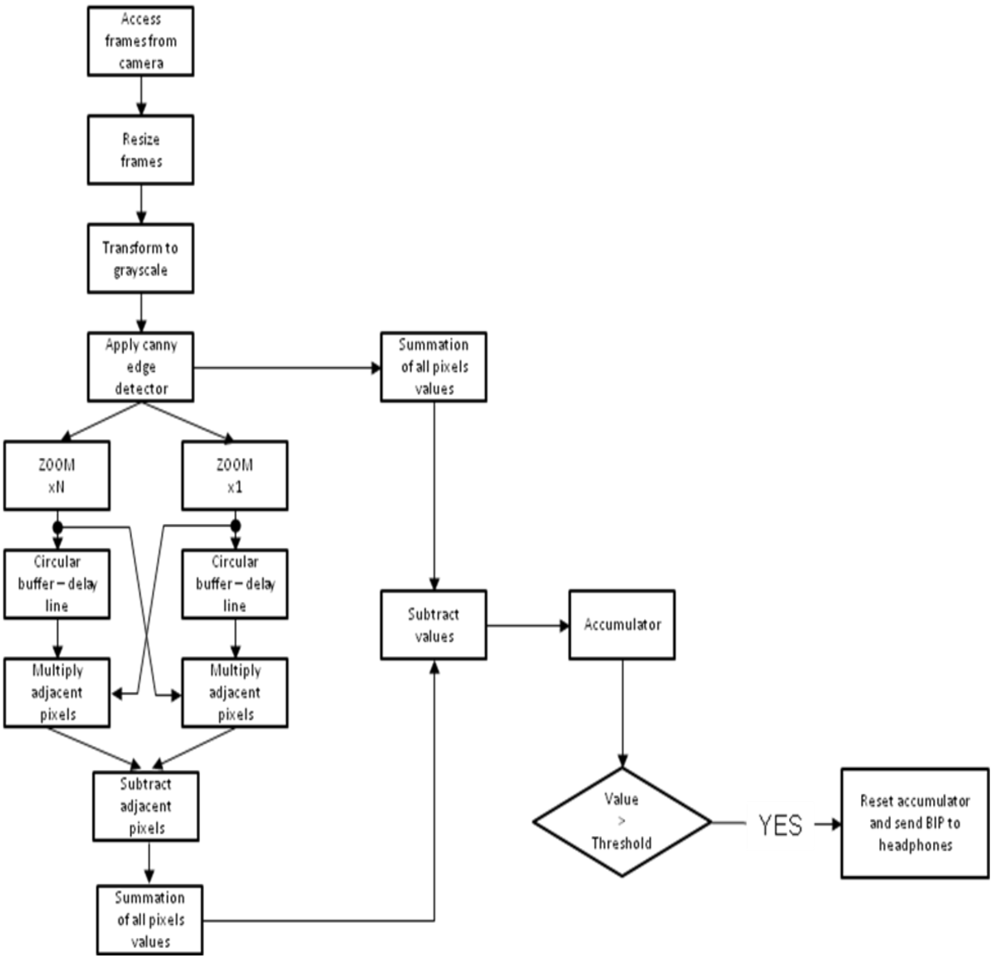

Access frames from camera
Resize frames
Transform to grayscale
Apply canny edge detector
Summation of all pixels values
ZOOM xN
ZOOM x1
Circular buffer – delay line
Circular buffer – delay line
Multiply adjacent pixels
Multiply adjacent pixels
Subtract adjacent pixels
Summation of all pixels values
Subtract values
Accumulator
Value > Threshold
YES
Reset accumulator and send BIP to headphones

Fig. 4.47. Block diagram of the software based implementation for the bio-inspired obstacle detection system (simplified version)

The frames received from the phone camera are taken and resized in order to save some speed processing (if necessary – depending on the technical specification of the particular smartphone used). To access the device camera, it must be declared the CAMERA permission in your Android Manifest. Also, there must be included the <uses-feature> manifest element to declare camera features used by application. For example, the camera and auto-focus feature are used, the Manifest should include the following:

```
<uses-permission android:name="android.permission.CAMERA" />
 <uses-feature android:name="android.hardware.camera" />
 <uses-feature android:name="android.hardware.camera.autofocus"
/>
```

The Interface Camera.PreviewCallback uses the public method:

onPreviewFrame(byte[] data, Camera camera)

with the following parameters [108]:

data -- the contents of the preview frame in the format defined by ImageFormat, which can be queried with getPreviewFormat(). If *setPreviewFormat(int)* is never called, the default will be the YCbCr_420_SP (NV21) format.

Camera-- the Camera service object.

After the frames are converted to grayscale using:

capture.retrieve(mGray, Highgui.CV_CAP_ANDROID_GREY_FRAME);

a canny edge detection is applied with *Imgproc.Canny().*

The last "n" processed frames are passed in two circular buffers and in one of the circular buffer the frames are enlarged (zoomed) using *mRgba.submat()* method. A new matrix is created after subtracting the adjacent pixels from the two circular buffers according with the block diagram. The value that corresponds with the summation of all elements from the matrix created is memorized and subtracted from the valuea that represents the summation of all pixel values from the reference frame. These values are subtracted between them and the result is passed into an accumulator. When the accumulator exceeds a threshold value, using the *soundPool.play*() method a sound is send to the headphones and the process starts again signaling a possible obstacle approaching.

The detailed software implementation (the source code) cand be seen in the ANNEX section.

The algorithm from Fig.4.47 it's a simplified version in comparison with the algorithm proposed in Chapter 3 that was simulated using Matlab/Simulink, and it can be used only in more simple environments (background that contains no edges and only one object that interfere in the trajectory of motion). In order to make a more accurate obstacle detection with this application it is necessary also the smarthphone camera to have an "optical image stabilization" incorporated.

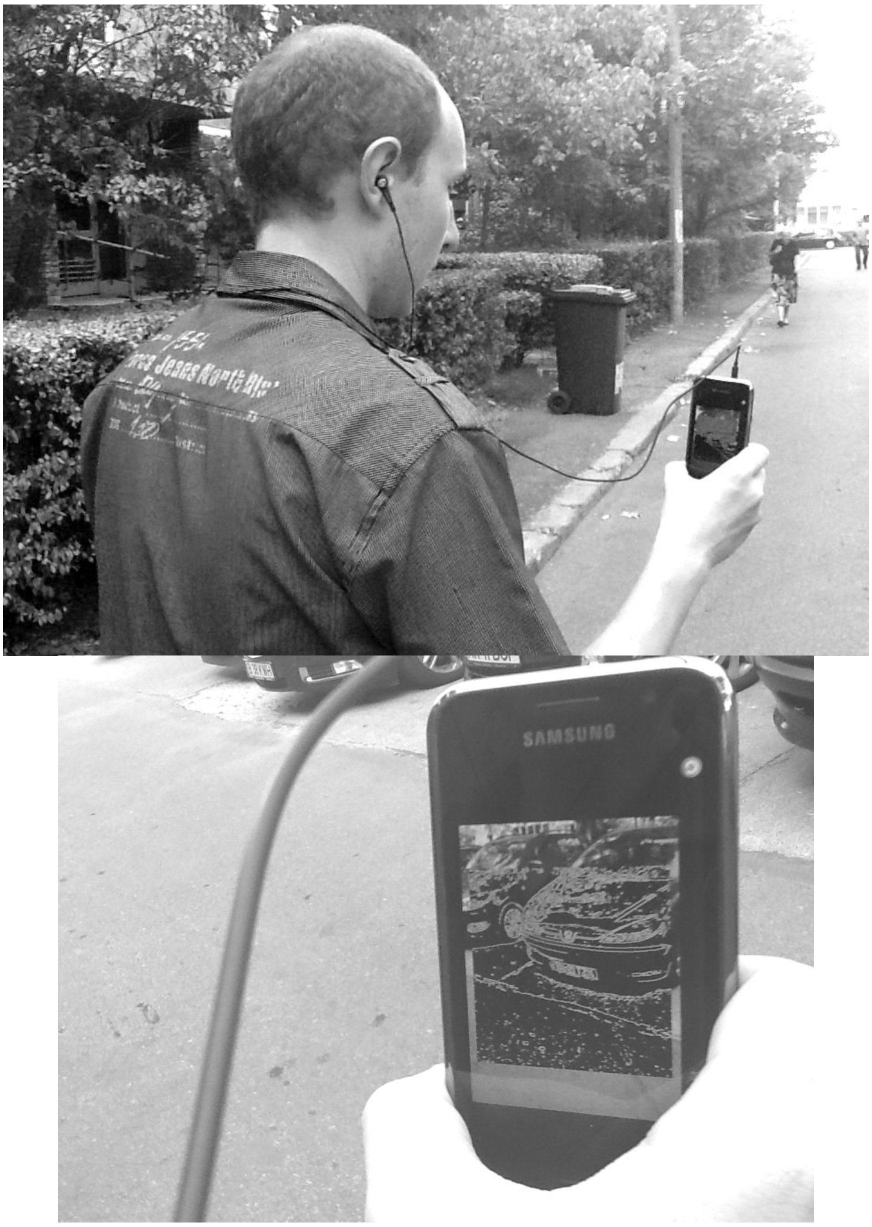


Fig. 4.48. Outdoor usage of the obstacle detection implemented as an application for a smartphone (obstacle in this case is a stationary car). When the person is approaching the car a beep is send to the headphones signaling the obstacle.

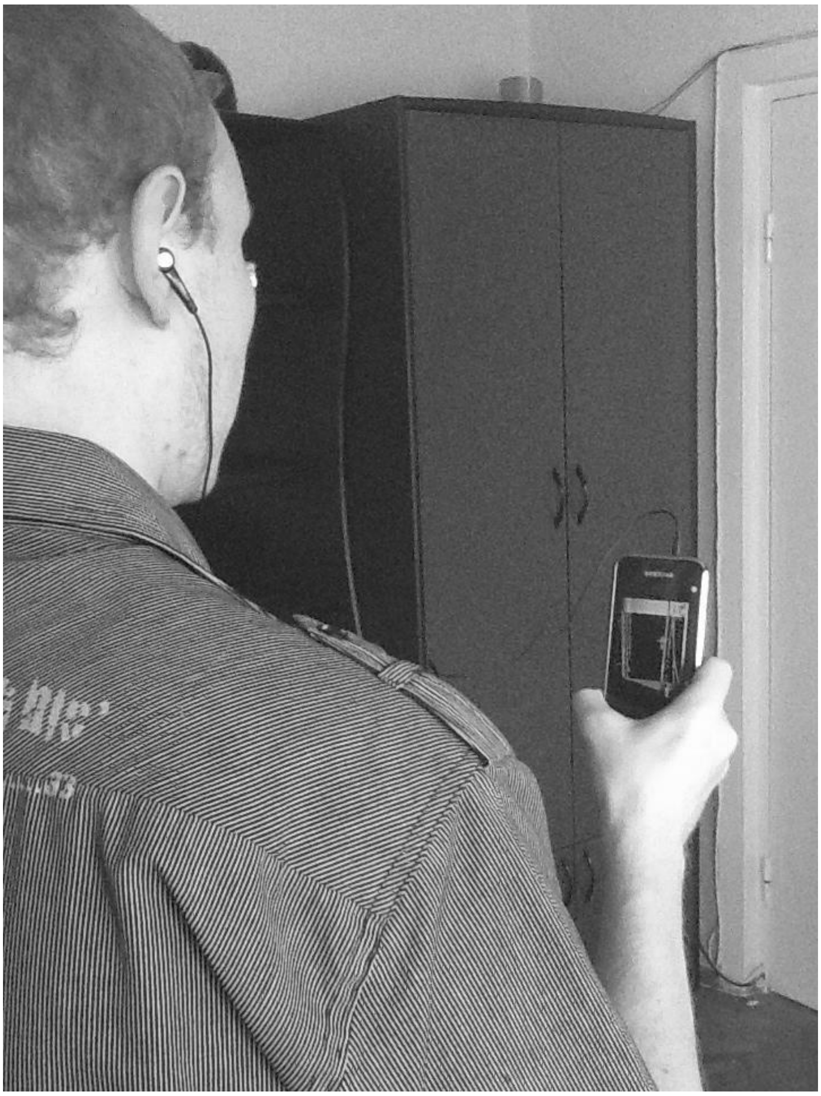


Fig. 4.49 Indoor usage of the obstacle detection implemented as an application for a smartphone (obstacle in this case is a closed door). When the person is approaching the closed door a beep is send to the headphones signaling the obstacle.

5 Conclusions and personal contributions

5.1 Conclusions

The present work is dedicated for the research of a bio-inspired obstacle detection system based on insect vision that can be used either as a guidance device for visually impaired people in order to replace the traditional tools like dogs or white cane, or as an additional safety assistive device for cars in automotive industry.

Based on the ideas and results presented in the thesis, the main conclusions are summarized as follows:

- In the last few years neuromorphic engineering has increased the interest and bio-inspired artificial systems even if are categorized as a niche domain their applicability represent a real interest.
- Choosing as inspiration the insect's vision in particular the fly and the locust has proved to be a good choice since the results are promising and their structure is much simpler to understand and implement.
- The existing bio-inspired obstacle detection systems on the market are still in the phase of "proof of concept" and are going through a development process in order to be used in a complex real environment which is very difficult to achieve yet.
- The proposed bio-inspired motion and obstacle detection systems had brought some improvements at the level of extra features like the motion detector with the possibility to detect the sense and the direction of moving obstacles and also at the level of concept for the bio-inspired obstacle detection in order to be used in a more complex environment.
- The solutions for the implementation of such a system were also analyzed from a theoretical point of view and evaluated through simulation using Matlab/Simulink and SPICE tools, and also from a practical side by integrating the algorithm in a smart-phone device.

Future research and development directions

Bio-inspired artificial systems are in a growing process. The author wishes to continue his research and development in this field evaluating new unexplored ideas at the level of concept and also at the level of implementation:

- New structures for the bio-inspired obstacle detection systems that can be used independently of the environment with minimum errors in detection.
- Practical implementation of the whole system into a single chip in order to increase the speed of processing, reduce costs as possible and power consumption for a higher autonomy.

- Increase the complexity of the algorithm used for the implementation of the system into a smart-phone device as technology evolution permits, knowing that in the last period the performance of these types of mobile phones has dramatically improved.

5.2 Original contributions

This section of the thesis presents the original contributions, from the author point of view.

Theoretical Contributions:

- An improved bio-inspired motion detection system concept using only one visual sensor and with extra features in order to be able to detect the sense and the direction of a moving obstacle.
- An enhanced bio-inspired obstacle detection system concept that uses only one visual sensor and extracts from the visual field only the parts that contain the information which is essential to detect an obstacle on the trajectory of motion in order to increase the speed of processing which is essential when it comes to be used in real time.
- A novel idea of using a fuzzy controller as a decision block for the bio-inspired obstacle detection system.

Applicative Contributions:

- An overview on existing bio-inspired motion and obstacle detection systems.
- Implementation of the bio-inspired motion and obstacle detection systems into a Matlab/Simulink model in order to evaluate the functionality and also to be used as a base for future development in this direction.
- Modeling the bio-inspired obstacle detection system concept at transistor level using TANNER tools (S-Edit, T-Spice) and evaluate the functionality and performance through simulation.
- Two novel structures for the bio-inspired obstacle detection system, one that uses voltage mode computational CMOS circuits and the other that uses current mode computational CMOS circuits, both resulting in more accurate results.
- An enhanced bio-inspired adaptive logarithmic photoreceptor structure that allows an increased gain for transient signals that contains the useful information, reduce the DC gain and allows synchronization between the processing part and the photoreceptor;
- A bio-inspired obstacle detection system structure with additional integrated temperature compensation that improves considerably the overall performance.
- A novel idea of implementing the algorithm of the bio-inspired obstacle detection system into a smart-phone device that has the advantage of virtually zero costs (if a person already is in possession of a smart-phone) and allows easy future improvements and development, also with no additional costs.

References

Author’s papers related to this Ph. D. thesis

[1] **Basch Mihai Emanuel**, Cristea D. G., Tiponut V., Slavici T. ”Elaborated Motion Detector based on Hassenstein-Reichardt correlator Model”, LATEST TRENDS on SYSTEMS, WSEAS, ISSN : 1792-4235, ISBN : 978-960-474-199-1, Corfu Island, Greece, July 22-24, 2010, pp.192-195;

[2] **Basch Mihai Emanuel**, Tiponut V., Ianchis D., Haraszy Z. ”Work Directions and New Results in Electronic Travel Aids for Blind and Visually Impaired People” LATEST TRENDS on SYSTEMS, WSEAS, ISSN : 1792-4235, ISBN : 978-960-474-199-1, Corfu Island, Greece,vol.II,22-24 Iulie 2010,pp.347-352;

[3] Cristea D.G, **Basch Mihai Emanuel**, Shacham-Diamand Yosi, Tiponut V., Hadar Ben-Yoav, Haraszy Z. ”VLSI Universal Signal Conditioning Circuit for Electrochemical and Bioluminescent Sensors”, Proc. 2010 IEEE 26-th Convention of Electrical and Electronics Engineers in Israel, ISBN: 978-1-4244-8680-9; IEEE Catalog Number: CFP10417-CDR; Eilat, Israel, 17-20 November 2010, pp.249-252;

[4] **Basch Mihai Emanuel**, Tiponut V., Ianchis D., Haraszy Z. ”Work Directions and New Results in Electronic Travel Aids for Blind and Visually Impaired People” WSEAS TRANSACTIONS on SYSTEM , Corfu Island, Greece, ISSN : 1109-2777, Issue 10, Volume 9 ,October 2010, pp.1086-1097;

[5] **Basch Mihai Emanuel**, Cristea D.G, Róbert I. L., ”A Bio-inspired Obstacle Avoidance System Concept for Visually Impaired People”, Recent Researches in System Science , WSEAS**,** ISSN: 1792-4235, ISBN:978-1-61804-023-7; Corfu Island, Greece, 14-16 July, 2011, pp.288-291;

[6] Cristea D.G, **Basch Mihai Emanuel**, Hadar Ben-Yoav, Virgil Tiponut, Yosi Shacham-Diamand, “Universal Signal Conditioning System for Amperometric Sensors”, Advances in Electrical and Computer Engineering, ISSN: 1582-7445, e-ISSN: 1844-7600, Suceava, Romania, Volume 12, Issue 1, Year 2012, pp. 19 – 24;

[7] **Basch Mihai Emanuel**, Róbert I. L., Cristea D.G, Virgil Tiponut, I. Bogdanov, “ A Bio-inspired Collision Avoidance System Concept for People with Visual Disabilities”, INTERNATIONAL JOURNAL OF SYSTEMS APPLICATIONS, ENGINEERING & DEVELOPMENT, Issue 6, Volume 5, October 2011, pp.701-709

[8] **Basch Mihai Emanuel**, “Obstacle detection concept based on biological visual systems”, Workshop nr.2 , „INTERDISCIPLINARITATEA SI MANAGEMENTUL CERCETARII” PREZENTAREA REZULTATELOR OBTINUTE DE DOCTORANZI, Nov. 2011, EL-37-EL-38

[9] **Basch Mihai Emanuel**, Róbert I. L., Cristea D.G, Virgil Tiponut, “Enhanced Bio-inspired Collision Avoidance System Concept with Fuzzy Logic Decision”, 13th INTERNATIONAL CONFERENCE ON OPTIMIZATION OF ELECTRICAL AND ELECTRONIC EQUIPMENT, Proc. 2012 IEEE, ISBN: 978-1-4673-1653-8; ISSN:1842-0133, Brasov, Romania, May 2012, pp.1525- 1530;

[10] **Basch Mihai Emanuel**, “Bio-Inspired Obstacle Detection System With Fuzzy Logic Decision”, Workshop nr.3 INTERDISCIPLINARITATEA SI MANAGEMENTUL CERCETARII IN STUDIILE DOCTORALE, 7-8 Iunie 2012 Oradea

Papers related to collaborations with other research teams not directly related to this Ph. D. thesis

[11]Róbert I. L., **Basch Mihai Emanuel**, Tiponut V., "Improved Power Distribution Method for BLDC Motor Driving Power Inverters" Proceedings of the IEEE 10th – ISSCS conference, June 2011; ISBN: 978-1-4577-0201-3 (IEEE Explore, INSPEC);

[12]Róbert I. L., **Basch Mihai Emanuel**, Bogdanov I., Tiponut V. Cristea D. " Hardware Implementation of Field-Weakening BLDC Motor Control" Proceedings of the 15th WSEAS International Conference on Circuits, July 2011; ISBN:978-1-61804-017-6;

[13]Róbert I. L., **Basch Mihai Emanuel**, Bogdanov I., Tiponut V. "Hardware Implementation of a BLDC Motor Diagnosis" Proceedings of the 15th WSEAS International Conference on Circuits, July 2011; ISBN:978-1-61804-017-6;

[14]Róbert I. L., **Basch Mihai Emanuel**, Bogdanov Ivan, Tiponut Virgil, Beschieru Adrian "Hardware Implementation of a BLDC Motor and Control System Diagnosis" International Journal of Circuits, Systems and Signal Processing, NAUN press, Issue 5, Volume 6, ISSN: 1998-4464, Oct., 2011;

[15]D. G. Cristea, **Basch Mihai Emanuel**, Z. Haraszy, N. Fishelson, V. Tiponut, "A new Electronic Equivalent Circuit for Electrochemical Cells", 2nd International Conference on Modelling and Simulations, ICOMOS 2011, ISSN:2038-3975, 11-25 Iulie 2011

[16]D. G. Cristea, **Basch Mihai Emanuel**, Z. Haraszy, V. Tiponut,"A new model for simulationg electrochemical sensors array", 2011 IEEE 17th International Symposium for Design and Technology in Electronic Packaging (SIITME)**,** 20-23 Octombrie 2011

References of the thesis

[17]Alexander Borst, " Drosophila's View on Insect Vision", Current Biology 19, R36–R47, January 13, 2009;

[18]Land, M.F., and Nilsson, D.-E. (2002). Animal Eyes (Oxford: Oxford University Press);

[19]Exner, S. (1891). Die Physiologie der facettierten Augen von Krebsen und Insecten: eine Studie. (Leipzig: Deuticke). English translation by RC Hardie: "The Physiology of the Compound Eyes of Insects and Crustaceans", (Springer), 1989;

[20]Exner, S. (1894). Entwurf zu einer physiologischen Erkla¨ rung der psychischen Erscheinungen (Leipzig, Wien: Deuticke);

[21]Kirschfeld, K. (1967). Die Projektion der optischen Umwelt auf das Raster der Rhabdomere im Komplexauge von MUSCA. Exp. Brain Res. 3, pp. 248–270;

[22]Coggshall, J.C., Boschek, C.B., and Buchner, S.M. (1973). Preliminary investigations on a pair of giant fibers in the central nervous system of dipteran flies. Z. Naturforsch. 28c, pp. 783–784;

[23]Strausfeld, N.J. (1976). Atlas of an Insect Brain (Berlin, Heidelberg Springer);

[24]Jaervilehto, M., and Zettler, F. (1971). Localized intracellular potentials from pre- and postsynaptic components in the external plexiform layer of an insect retina. Z. vergl. Physiol. 75,pp. 422–440;

[25]Straka, H., and Ammermu¨ ller, J. (1991). Temporal resolving power of blowfly visual system: effects of decamethonium and hyperpolarization on responsesof laminar monopolar neurons. J. Comp. Physiol. A 168, pp. 129–139;

[26]Takemura, S.Y., Lu, Z., and Meinertzhagen, I.A. (2008). Synaptic circuits of the Drosophila optic lobe: the input terminals to the medulla. J. Comp. Neurol. 509, 493–513;

[27]von Frisch, K. (1947). The Dance Language and Orientation of Bees (Cambridge: Harvard University Press);

[28]Rossel, S., and Wehner, R. (1986). Polarization vision in bees. Nature 323, 128-131;

[29]von Philipsborn, A., and Labhart, T. (1990). A behavioral study of polarization vision in the fly, Musca domestica. J. Comp. Physiol. A 167, 737–743;

[30]Dacke, M., Nordstrom, P., and Scholtz, C.H. (2003). Twilight orientation to polarised light in the crepuscular dung beetle Scarabaeus zambesianus, J. Exp. Biol. 206, 1535–1543;

[31]Voss, R., and Zeil, J. (1998). Active vision in insects: an analysis of objectdirected zig-zag flights in wasps (Odynerus spinipes, Eumenidae). J. Comp. Physiol. A 182, 373–387;

[32]Sobel, E.C. (1990). Depth perception by motion parallax and paradoxical parallax in the locust. Naturwiss. 77, 241–243;

[33]Gilbert, C. (1997). Visual control of cursorial prey pursuit by tiger beetles (Cicindelidae). J. Comp. Physiol. A 181, 217–230;

[34]Olberg, R.M., Worthington, A.H., and Venator, K.R. (2000). Prey pursuit and interception in dragonflies. J. Comp. Physiol. A 186, pp.155–162;

[35]Borst, A., and Bahde, S. (1988). Spatio-temporal integration of motion - A simple strategy for safe landing in flies. Naturwiss. 75, pp.265–267;

[36]Hatsopoulos, N., Gabbiani, F., and Laurent, G. (1995). Elementary computation of object approach by a wide-field visual neuron. Science 270, pp.1000–1003;

[37]Rind, F.C., and Simmons, P.J. (1997). Signaling of object approach by the DCMD neuron of the locust. J. Neurophysiol. 77,pp. 1029–1033;

[38]Borst, A., and Egelhaaf, M. (1989). Principles of visual motion detection. Trends Neurosci. 12, pp. 297–30;

[39]Reichardt, W. (1961). Autocorrelation, a principle for the evaluation of sensory information by the central nervous system. In Sensory Communication, W.A. Rosenblith, ed. (New York, London: The M.I.T. Press and John Wiley & Sons), pp. 303–317;

[40]Reichardt, W. (1987). Evaluation of optical motion information by movement detectors. J. Comp. Physiol. A 161,pp. 533–547;

[41]Haag, J., Vermeulen, A., and Borst, A. (1999). The intrinsic electrophysiological characteristics of fly lobula plate tangential cells: III. Visual response properties. J. Computat. Neurosci. 7, 213–234;

[42]Roger D. Santer, Yoshifumi Yamawaki, F. Claire Rind, Peter J. Simmons, "Preparing for escape: an examination of the role of the DCMD neuron in locust escape jumps", J Comp Physiol A (2008) Vol. 194, pp.69–77;

[43]F. Claire Rind and Peter J. Simmons," Seeing what is coming: building collision-sensitive neurons", Trends Neurosci. (1999), Vol.22, pp.215–220;

[44]Judge, S. and Rind, F.C. (1997) J. Exp. Biol. 200, pp. 2209–2216;

[45]Rowell, C.H.F. (1971) Z. Vergl. Physiol. 73, 167–194;

[46]Young, D. (1989) Nerve Cells and Animal Behaviour, Cambridge University Press;

[47]Rind, F.C. and Simmons, P.J. (1997) J. Neurophysiol. 77, pp. 1029–1033;

[48]Rind, F.C. (1984) J. Exp. Biol. 110, 143–167;

[49]Rind, F.C. and Simmons, P.J. (1992) J. Neurophysiol. 68, pp. 1654–1666;

[50]Hatsopoulos, N., Gabbiani, F. and Laurent, G. (1995) Science 270, pp. 1000–1003;

[51]Rind, F.C. (1996) J. Neurophysiol. 75,pp. 986–995;

[52]Rind, F.C. and Bramwell, D.I. (1996) J. Neurophysiol. 75, pp. 967–985;

[53]Patrick A. Shoemakera, David C. O'Carrollb, Andrew D. Strawb, "Implementation of visual motion detection with contrast adaptation", Proceedings of SPIE Vol. 4591, 2001, pp. 316–327;

[54]Warzecha, A.K., Egelhaaf, M., and Borst, A. "Neural circuit tuning fly visual interneurons to motion of small objects. 1. Dissection of the circuit by pharmacological and photoinactivation techniques," J. Neurophysiol. Vol. 69, pp. 329-339, 1993;

[55]Olberg, R.M., Worthington, A.H. & Venator, K.R. "Prey pursuit and interception in dragonflies," J. Comp. Physiol. A vol. 186, pp. 155-162, 2000;

[56]M. O'Shea, J. L. D. Williams, The anatomy and output connection of a locust visual interneurone; the lobular giant movement detector (LGMD) neurone" J. Comparative Physiology, vol. 91, 1974, pp. 257- 266;

[57]F. C. Rind and P. J. Simmons, "Orthopteran DCMD neuron: a reevaluation of responses to moving objects. I. Selective responses to approaching objects," J. Neurophys., vol. 68, 1992, pp. 1654– 1666;

[58] F. C. Rind, D. I. Bramwell, Neural network based on the input organization of an identified neuron signaling impending collision, Journal of Neurophysiology, Vol 75, Issue 3, 1996, pp. 967-985;

[59] G. Linan-Cembrano, L. Carranza, C. Rind, A. Zarandy, M. Soininen, A. Rodriguez-Vazquez, Insect-vision inspired collision warning vision processor for automobiles, Circuits and Systems Magazine, IEEE, Volume 8, Issue 2, 2008, pp. 6-24;

[60] Bermúdez, i. Badia, S., and Verschure, A collision avoidance model based on the Lobula Giant Movement Detector (LGMD) neuron of the locust, In Proceedings of International Joint Conference on Neural Networks (IJCNN'04), 2004;

[61] Douglas, J. K. and Strausfeld, N. J., Visual motiondetection in flies: parallel direction and non-direction sensitive pathways between the medulla and lobula plate, The Journal of Neuroscience, 1996, 16(15): 4551–4562;

[62] Sergi Bermudez i Badia, Pawel Pyk and Paul F.M.J. Verschure, A fly-locust based neuronal control system applied to an unmanned aerial vehicle: the invertebrate neuronal principles for course stabilization, altitude control and collision avoidance, The International Journal of Robotics Research 2007, 26, 759;

[63] Hatsopoulus, N., Gabbiani, F. & Laurent, G., Elemntary computation of object approach by a wide-field visual neuron, Science 270, 1995, pp. 1000-1003;

[64] Jean-Christophe Zufferey, Dario Floreano, Matthijs van Leewen, Tancredi Merenda, Evolving Vision- Based Flying Robots, In Proceedings of the 2nd International Workshop on Biologically Motivated Computer Vision, LNCS 2525, 2002, Berlin, Springer-Verlag, pp. 592-600;

[65] B. Hassenstein and W. Reichardt, Structure of a mechanism of perception of optical movement, Proceedings of the 1st International Conference on Cybernetics, 1956, pp. 797–801;

[66] Christian Lutkemyes and Tobias Noll, "A Transversal Equalizer with an Increased Adoptive Speed Avid Capability." IEEE Journal of Solid State Circuits 33(3), pp. 503–507, 1998;

[67] XiaodongWang and Richard Spenser, "A Low-Power 170 MHz Discrete Time Analog FIR Filter." IEEE Journal of Solid State Circuits 33(3), pp. 417–426, 1998;

[68] Gabriel Gomez and Raymond Siferd, "An Adaptive Noise Canceller Implemented with CMOS Analog Technology." Journal of Circuits Systems and Computers 6(2), pp. 139–154, 1996;

[69] David A. Martin, Hae-Seung Lee, and Ichiro Masaki, "AMixed- Signal Array Processor with Early Vision Applications." IEEE Journal of Solid State Circuits 33(3), pp. 497–502, 1998;

[70] Shuo-Yuan Hsiao and Chung-Yu Wu, "A Parallel Structure for CMOS Four Quadrant Analog Multipliers and its Application to a 2 GHz RF Downconversion Mixer." IEEE Journal of Solid State Circuits 33(6), pp. 859–869, 1998;

[71] Gian Marco Bo, Daniel D. Caviglia, and Maurizio Valle, "An Analog VLSI Implementation of a Feature Extractor for Real- Time Optical Character Recognition." IEEE Journal of Solid State Circuits 33(4), pp. 556–564, 1998;

[72] Delbrück, T. and Mead, C. A. Analog VLSI adaptive logarithmic wide-dynamic-range photoreceptor. vol.4, 339-342. 1994. London. 1994 IEEE International Symposium on Circuits and Systems;

[73] Kramer, J., "An integrated optical transient sensor," IEEE Trans. on Circuits and Systems II, vol. 49, no. 9, pp. 612-628, Sept.2002;

[74] Liu, S. C. Silicon retina with adaptive filtering properties. Jordan, M. I., Kearns, M. J., and Solla, S. A. vol.10, 712-718. 1998. MIT Press. Advances in Neural Information Processing Systems 10. Jordan, M. I., Kearns, M. J., and Solla, S. A;

[75] Jing Xu, Ray Siferd, Robert L. Ewing, "High Performance CMOS Analog Arithmetic Circuits", Analog Integrated Circuits and Signal Processing, (1999) Vol.22, pp.193–201;

[76] D.A. Baylor, G. Matthews, K.-W. Yau, "Two components of electrical dark noise in toad rod outer segments," J. Physiol. vol. 309, pp. 591–621, 1980;

[77] R. G. Benson and T. Delbrück, (1991). Direction-selective silicon retina that uses null inhibition, in D.S. Touretzky, Ed. Advances in Neural Information Processing Systems 4. pp. 756–763;

[78] W.C. Dash and R. Newman, "Intrinsic optical absorption in single-crystal germanium and silicon at 77 °K and 300 °K," Physical Review, vol. 99, pp. 1151–1155, 1955;

[79]D. Cohen and G. Lewicki, "MOSIS—the ARPA silicon broker," in Proceedings from the Second Caltech Conference on VLSI, California Inst. of Tech., Pasadena CA, pp. 29-44, 1981;

[80]T. Delbrück, "Silicon retina with Correlation- Based, Velocity-Tuned Pixels," IEEE Transactions on Neural Networks, vol. 4, no. 3, pp. 529–541, May 1993;

[81]T. Delbrück, Investigations of Analog VLSI Phototransduction and Visual Motion Processing, Ph.D. Thesis, Dept. of Computation and Neural Systems, California Institute of Technology, Pasadena, CA, 91125, 1993;

[82]Reid R. Harrison, Christof Koch, "A Robust Analog VLSI Reichardt Motion Sensor", Analog Integrated Circuits and Signal Processing,(200) Vol. 24, pp.213-229;

[83]T. Delbruck ,D. Oberhoff, "SELF-BIASING LOW POWER ADAPTIVE PHOTORECEPTOR", Inst. of Neuroinformatics, ETH/Univ. Zürich, Switzerland, Press release

[84]T. Delbrück and C.A. Mead, (1991), Silicon adaptive photoreceptor array that computes temporal intensity derivatives, in T.S. Jay Jayadev (ed.) Proc. SPIE, Infrared Sensors: Detectors, Electronics, and Signal Processing., vol. 1541, pp 92–99;

[85]J. Choi, B. J. Sheu, and J. C.-F. Chang, "A Gaussian synapse circuit for analog VLSI neural circuits," IEEE Trans. VLSI Syst., vol. 2, pp. 129-133, Mar. 1994;

[86]K. Bult and H. Wallinga, "A class of analog CMOS circuits based on the square-law characteristic of an MOS transistor in saturation," IEEE J. Solid-State Circuits, vol. SC-22, pp. 357-365, June 1987;

[87]T. Delbrück, C.A. Mead," ANALOG VLSI PHOTOTRANSDUCTION by continuous-time, adaptive, logarithmic photoreceptor circuits", CNS Memo No. 30, April 2, 1996;

[88]R. J. Widlar, "Some circuit design & techniques for linear integrated circuits," IEEE Trans. Circuit Theory. vol. CT-12. DD. 586.590. Dec.1965;

[89]E. A. Vittoz, "The design of high-performance analog circuits on digital CMOS chips," IEEE J. Solid-State Circuits, vol. SC-20, DD. 948-955 June 1985;

[90]M. J. M. Pelgrom et al., "Matching properties of MOS transistors," IEEE J. Solid-State Circuits, vol. 24, pp. 1433-1440, Oct. 1989;

[91]K. Bult and H. Walliinga, "A class of analog CMOS circuits based on the square-law characteristic of an MOS transistor in saturation," IEEE I. Solid-state Circuits, Vol. SC-22, pp. 357-365, June 1987;

[92]J.S. Pena-Fino1 and J.A. Connelly, "A MOS four-quadrant analog multiplier using the quarter-square technique,"I EEE J. Solid-state Circuits, Vol. SC-22, pp. 10641073, Dec. 1987;

[93]H. Song and C. Kim, "An MOS four-quadrant analog multiplier using simple two-input squaring circuits with source followers," IEEE 1. Solid-State Circuits, Vol. SC-25. pp. 841-847, June 1990;

[94]E. Seevinck and RJ. Wiegerinlt, "Generalized translinear circuit principle," accepted for publication in the IEEE I. Solid-state Cimits;

[95]J. Madrenas, M. Verleysen, P. Thissen, J. L. Voz, "A CMOS Analog Circuit for Gaussian Functions", IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-II: Analog And Digital Signal Processing.Vol.43, No. I, January 1996

[96]A. Cichocki and R. Unbehauen, 'Technique for compensation of errors in analogue multipliers," Elecbmics Letters, Vol. 25, pp. 305-307, March 1989;

[97]Rind, F.C. and Bramwell, D.I. (1996) J. Neurophysiol. 75, 967–985.

[98]Kopal Gupta, Prof. B. P Singh, Rockey Choudhary," Design and Analysis of Current-to-Voltage and Voltage -to-Current Converters using 0.35µm technology", International Journal of Emerging Technology and Advanced Engineering, Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 4, April 2012)

[99]Remco J. Wiegerink, "A CMOS Four-Quadrant Analog Current Multiplier", MESA Rescarch Institute, Twente University, Press release

[100] Chun Wei Lin, Yu Huan Wu, Sheng Feng Lin, "A Precise Current Subtractor Design", 2011 International Conference on Circuits, System and Simulation, IPCSIT vol.7 (2011) © (2011) IACSIT Press, Singapore

[101] Kazuki Nakada, Tetsuya Asai, Tetsuya Hirose, Yoshihito Amemiya, "Analog Current-Mode CMOS Implementation of Central Pattern Generator for Robot Locomotion", Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, July 31 - August 4, 2005

[102] B. Hassenstein andW. Reichardt, ``Systemtheoretische Analyse der Zeit-, Reihenfolgen-, und Vorzeichenauswertung bei der Bewegungsperzeption des RuÈsselkaÈfers Chlorophanus.'' Z. Naturforch. 11b, (1956), pp. 513-524;

[103] H. B. Barlow and W. R. Levick, ``The mechanism of directionally selective units in the rabbit's retina.'' Journal of Physiology, Vol. 178, (1965), pp. 447-504;

[104] A. Borst and M. Egelhaaf, ``Principles of visual motion detection.'' Trends in Neuroscience Vol. 12, (1989), pp. 297-306;

[105] Reid R. Harrison, "A Biologically Inspired Analog IC for Visual Collision Detection", IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, Vol. 52, No. 11, Nov. 2005.

[106] Cristea D.G., "UNIVERSAL SIGNAL CONDITIONING SYSTEM FOR ELECTROCHEMICAL AND BIOLUMINESCENT SENSOR ARRAYS", PhD. Thesis, Politehnica – Timişoara, 2011.

[107] Smartphone , Available from: http://en.wikipedia.org/wiki/Smartphone

[108] http://developer.android.com/reference/android/hardware/Camera.html

ANNEX

SPICE NETLIST for each schematic use in this thesis (include simulation settings in T-SPICE also when is the case)

One element of the adaptive photoreceptor with the voltage mode computational circuits and temperature compensation circuit

```
**** Simulation Settings - General section****

.lib"C:\Tanner Tools v13.0\Libraries\Models\Generic_025.lib"

*************** Subcircuits *****************
.subckt A0_Real InN InP Out Gnd Vdd
*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 Out N_3 N_4 N_4 NMOS W=35u L=2.5u AS=31.5p PS=71.8u AD=31.5p PD=71.8u
MNMOS_2 N_1 InN N_2 Gnd NMOS W=15u L=2.5u AS=13.5p PS=31.8u AD=13.5p PD=31.8u
MNMOS_3 N_3 InP N_2 Gnd NMOS W=15u L=2.5u AS=13.5p PS=31.8u AD=13.5p PD=31.8u
MNMOS_4 N_2 Vdd N_4 N_4 NMOS W=80u L=2.5u AS=72p PS=161.8u AD=72p PD=161.8u
MPMOS_1 Out N_3 Vdd Vdd PMOS W=75u L=2.5u AS=67.5p PS=151.8u AD=67.5p PD=151.8u
MPMOS_2 N_1 N_1 Vdd Vdd PMOS W=30u L=2.5u AS=27p PS=61.8u AD=27p PD=61.8u
MPMOS_3 N_3 N_1 Vdd Vdd PMOS W=30u L=2.5u AS=27p PS=61.8u AD=27p PD=61.8u
VVoltageSource_1 N_4 Gnd  DC -3.3
.ends

.subckt Adder In1 In2 Out Gnd Vdd Vss
*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 N_1 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
MNMOS_2 N_1 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
MNMOS_11 N_2 In1 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
MNMOS_3 N_3 N_3 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MNMOS_12 N_7 N_7 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
```

```
   MNMOS_4 N_4 N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_13 Out N_7 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_5 N_6 N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_14 N_2 In2 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_6 Out N_3 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_2 N_3 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_3 N_4 Gnd Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_4 N_4 Gnd Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_5 N_6 N_6 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_6 Out N_6 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_11 N_7 In1 Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_12 N_7 In2 Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_13 Out N_2 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_14 N_2 N_2 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   .ends

   .subckt AMP InN InP Out Gnd
   *-------- Devices: SPICE.ORDER > 0 --------
   EVCVS_E_Element_1 Out Gnd InP InN  100k
   .ends

   .subckt Current_Comparator InN InP Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_4 N_4 Gnd Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_2 Out InN N_3 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_3 N_3 N_4 Gnd Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_4 InN InN N_4 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
   MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
```

```
   MPMOS_3 Out InP N_2 Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
   MPMOS_4 InP InP N_1 Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
   .ends

   .subckt INV In Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 Out In Gnd Gnd NMOS W=200u L=1u AS=180p PS=401.8u AD=180p
PD=401.8u
   MPMOS_1 Out In Vdd Vdd PMOS W=500u L=1u AS=450p PS=1.0018m
AD=450p PD=1.0018m
   .ends

   .subckt INV4 In Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 In Gnd Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_2 N_2 N_1 Gnd Gnd NMOS W=20u L=1u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_3 Out N_2 Gnd Gnd NMOS W=40u L=1u AS=36p PS=81.8u AD=36p
PD=81.8u
   MPMOS_1 N_1 In Vdd Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p
PD=51.8u
   MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=50u L=1u AS=45p PS=101.8u AD=45p
PD=101.8u
   MPMOS_3 Out N_2 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   .ends

   .subckt Inverter_adder In1 In2 Out Gnd Vdd Vss
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_10 N_3 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_1 N_1 In2 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_2 N_1 In1 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_3 N_2 N_2 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_4 N_4 N_4 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_5 N_6 N_4 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_6 Out N_2 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_7 Out N_8 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_8 N_8 N_8 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
```

```
   MNMOS_9 N_3 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_3 N_4 In1 Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_4 N_4 In2 Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_5 N_6 N_6 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_6 Out N_6 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_7 Out N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_8 N_8 Gnd Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MPMOS_9 N_8 Gnd Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MPMOS_10 N_3 N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   .ends

   .subckt Multiplier4cadran In1 In2 In3 In4 Out Vdd Vss
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 In1 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_2 N_1 In2 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_3 N_2 N_2 Vss Vss NMOS W=80u L=8u AS=72p PS=161.8u AD=72p
PD=161.8u
   MNMOS_4 Out N_2 Vss Vss NMOS W=80u L=8u AS=72p PS=161.8u AD=72p
PD=161.8u
   MNMOS_5 N_3 In3 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_6 N_3 In4 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p
PD=161.8u
   MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p
PD=161.8u
   MPMOS_3 Out N_3 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p
PD=161.8u
   MPMOS_4 N_3 N_3 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p
PD=161.8u
   .ends

   .subckt MUX_2to1 CTRL In1 In2 Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 Out CTRL In2 Gnd NMOS W=5u L=1u AS=4.5p PS=11.8u AD=4.5p
PD=11.8u
   MNMOS_3 N_2 CTRL Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_4 N_2 In1 N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_5 N_2 Out N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_7 Out N_2 In1 Gnd NMOS W=5u L=1u AS=4.5p PS=11.8u AD=4.5p
PD=11.8u
   MNMOS_8 N_2 Out N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_9 N_2 In2 N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_1 Out N_2 In2 Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p
PD=51.8u
   MPMOS_3 N_2 CTRL Vdd Vdd PMOS W=50u L=1u AS=45p PS=101.8u AD=45p
PD=101.8u
   MPMOS_4 CTRL Out CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_5 CTRL In1 CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_7 Out CTRL In1 Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p
PD=51.8u
   MPMOS_8 CTRL In2 CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_9 CTRL Out CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   .ends

   .subckt Substractor In1 In2 Out Vdd Vss
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 N_1 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_2 N_2 N_1 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_3 Out N_6 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_4 Out N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_5 N_5 N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_6 N_4 In1 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_7 N_3 In2 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_8 N_6 N_6 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_1 N_1 In1 Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
```

```
   MPMOS_2 N_2 N_2 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_3 Out N_2 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_4 Out N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_5 N_5 N_4 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_6 N_4 N_4 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_7 N_3 N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_8 N_6 In2 Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   .ends

   .subckt Multiplier IN_V1 IN_V2 Out Gnd Vdd Vss
   XInverter_adder_1 IN_V1 IN_V2 N_8 Gnd Vdd Vss Inverter_adder
   XSubstractor_1 IN_V1 IN_V2 N_2 Vdd Vss Substractor
   XSubstractor_2 IN_V2 IN_V1 N_1 Vdd Vss Substractor
   XMultiplier4cadran_1 N_1 N_2 N_3 N_8 Out Vdd Vss Multiplier4cadran
   XAdder_1 IN_V1 IN_V2 N_3 Gnd Vdd Vss Adder
   .ends

   .subckt SW_IN In Out SEL Gnd Vdd
   *-------- Devices: SPICE.ORDER == 0 --------
   XINV_1 SEL N_1 Gnd Vdd INV
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 Out SEL In Gnd NMOS W=300u L=1u AS=270p PS=601.8u AD=270p
PD=601.8u
   MPMOS_1 Out N_1 In Vdd PMOS W=350u L=1.2u AS=315p PS=701.8u
AD=315p PD=701.8u
   .ends


   ********* Simulation Settings - Parameters and SPICE Options *********

   *-------- Devices: SPICE.ORDER == 0 --------
   XCurrent_Comparator_1 N_18 N_9 N_8 Gnd Vdd Current_Comparator
   XAMP_1 N_34 Gnd INput1 Gnd AMP
   XMultiplier_1 CH2 CH1_ZOOM_DELAYED N_24 Gnd Vdd Vss Multiplier
   XAMP_2 N_17 Gnd INput2 Gnd AMP
   XMultiplier_2 CH1_zoom CH2_DELAYED N_22 Gnd Vdd Vss Multiplier
   XSW_IN_2 N_11 N_17 N_12 Gnd Vdd SW_IN
   XA0_Real_3 CH2 HPF2 CH2 Gnd Vdd A0_Real
   XINV4_1 N_7 N_12 Gnd Vdd INV4
   XA0_Real_2 N_20 N_25 CH1_zoom Gnd Vdd A0_Real
   XA0_Real_7 HPF1 N_32 HPF1 Gnd Vdd A0_Real
   XSubstractor_1 N_24 N_22 Out Vdd Vss Substractor
   XMUX_2to1_1 N_8 N_10 N_21 N_33 Gnd Vdd MUX_2to1
   XA0_Real_1 HPF2 N_30 HPF2 Gnd Vdd A0_Real
   XSW_IN_1 N_11 N_6 N_7 Gnd Vdd SW_IN
   *-------- Devices: SPICE.ORDER > 0 --------
   CCapacitor_7 N_32 INput1  500f
   CCapacitor_3 N_30 INput2  500f
   CCapacitor_2 CH2_DELAYED Gnd  100n
   CCapacitor_4 CH1_ZOOM_DELAYED Gnd  100n
   RResistor_1 N_25 HPF1  R=500
   RResistor_2 N_20 CH1_zoom  R=1k
   RResistor_3 Gnd N_20  R=1k
   RResistor_4 CH1_zoom CH1_ZOOM_DELAYED  R=10meg
   RResistor_5 CH2 CH2_DELAYED  R=10meg
   RResistor_7 N_34 INput1  R=100k
   RResistor_6 N_17 INput2  R=100k
   MNMOS_1 N_5 N_2 N_4 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_2 N_1 N_3 Gnd Gnd NMOS W=15u L=2u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_3 N_15 N_15 N_16 Gnd NMOS W=60u L=1u AS=54p PS=121.8u
AD=54p PD=121.8u
   MNMOS_4 N_14 N_15 N_13 Gnd NMOS W=60u L=1u AS=54p PS=121.8u
AD=54p PD=121.8u
   MNMOS_5 Vdd N_1 N_13 Gnd NMOS W=25u L=1u AS=22.5p PS=51.8u
AD=22.5p PD=51.8u
   MNMOS_6 N_13 TEMP_CTLR Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_7 N_16 TEMP_CTLR Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_8 N_2 Vdd N_2 N_2 NMOS W=80u L=2u AS=72p PS=161.8u AD=72p
PD=161.8u
   MNMOS_9 N_1 N_2 N_1 N_1 NMOS W=2.5u L=2u AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_13 TEMP_CTLR N_27 Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_14 N_27 N_26 Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_15 N_28 Vdd Gnd Gnd NMOS W=25u L=1u AS=22.5p PS=51.8u
AD=22.5p PD=51.8u
   MNMOS_16 N_26 TEMP_CTLR N_28 Gnd NMOS W=15u L=1u AS=13.5p
PS=31.8u AD=13.5p PD=31.8u
   MNMOS_17 N_23 N_29 N_28 Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MPMOS_13 TEMP_CTLR N_27 Vdd Vdd PMOS W=40u L=1u AS=36p PS=81.8u
AD=36p PD=81.8u
   MPMOS_14 N_27 N_26 Vdd Vdd PMOS W=40u L=1u AS=36p PS=81.8u AD=36p
PD=81.8u
   MPMOS_1 N_21 N_5 Vdd Vdd PMOS W=100u L=2u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_2 N_2 N_1 N_2 N_2 PMOS W=2.5u L=2u AS=2.25p PS=6.8u AD=2.25p
PD=6.8u
   MPMOS_3 Vdd N_2 Vdd Vdd PMOS W=80u L=2u AS=72p PS=161.8u AD=72p
PD=161.8u
```

```
   MPMOS_4 N_1 N_1 N_2 N_2 PMOS W=2.5u L=50u AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_5 N_5 N_5 Vdd Vdd PMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MPMOS_6 N_1 N_5 Vdd Vdd PMOS W=200u L=2u AS=180p PS=401.8u
AD=180p PD=401.8u
   MPMOS_7 N_10 N_5 Vdd Vdd PMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MPMOS_8 N_26 N_23 Vdd Vdd PMOS W=35u L=1u AS=31.5p PS=71.8u
AD=31.5p PD=71.8u
   MPMOS_9 N_23 N_23 Vdd Vdd PMOS W=35u L=1u AS=31.5p PS=71.8u
AD=31.5p PD=71.8u
   DDiode_1 N_29 N_19 diode AREA=1 L=0 W=0
   DDiode_2 N_19 N_31 diode AREA=1 L=0 W=0
   DDiode_3 N_31 Gnd diode AREA=1 L=0 W=0
   VVss Vss Gnd  DC 3.3
   VVdd Vdd Gnd  DC 3.3
   VVoltageSource_1 N_7 Gnd  PULSE(0 3.3 0 150m 150m 19.7 40)
   ICurrentSource_2 N_9 Gnd  DC 100n
   ICurrentSource_3 Vdd N_15  DC 100n
   ICurrentSource_4 Vdd N_29  DC 500u
   ICurrentSource_1 N_3 Gnd  PULSE(10n 100n 0 150m 150m 19.7 40)
   .PRINT TRAN V(INput2)
   .PRINT TRAN V(N_7)
   .PRINT TRAN V(INput1)
   .PRINT TRAN V(HPF1)
   .PRINT TRAN V(HPF2)
   .PRINT TRAN V(CH1_ZOOM_DELAYED)
   .PRINT TRAN V(CH2_DELAYED)
   .PRINT TRAN V(CH1_zoom)
   .PRINT TRAN V(CH2)
   VAmmeter_1 N_4 N_3 0v
   .PRINT TRAN I(VAmmeter_1)
   VAmmeter_2 N_11 N_14 0v
   .PRINT TRAN I(VAmmeter_2)
   VAmmeter_3 N_34 N_6 0v
   .PRINT TRAN I(VAmmeter_3)

   ********* Simulation Settings - Analysis section *********
   .op
   .tran 0.05s 300s

   ********* Simulation Settings - Additional SPICE commands *********
   .option vmax=0

   .model diode D
   + is=18.8n rs=0.1 bv=2 ibv=5u cjo=30 m=0.333 n=2

   *.model diode1 D
   *+ is=4e-13 rs=1e12 bv=60 ibv=5u cjo=60p m=0.5 n=1.35
```

```
.param m=0


.end
```

One element of the adaptive photoreceptor with the voltage mode computational circuits and temperature compensation circuit

```
**** Simulation Settings - General section ****
.lib"C:\Tanner Tools v13.0\Libraries\Models\Generic_025.lib"

*************** Subcircuits *****************
.subckt Current_Comparator InN InP Out Gnd Vdd
*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 N_4 N_4 Gnd Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p PD=41.8u
MNMOS_2 Out InN N_3 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p PD=41.8u
MNMOS_3 N_3 N_4 Gnd Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p PD=41.8u
MNMOS_4 InN InN N_4 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p PD=41.8u
MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p PD=141.8u
MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p PD=141.8u
MPMOS_3 Out InP N_2 Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p PD=141.8u
MPMOS_4 InP InP N_1 Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p PD=141.8u
.ends

.subckt Current_LPF In Out Gnd Vdd
*-------- Devices: SPICE.ORDER > 0 --------
CCapacitor_2 N_2 Gnd  1u
MNMOS_7 N_1 N_1 N_2 Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p PD=121.8u
MNMOS_8 N_9 N_9 N_4 Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p PD=121.8u
MNMOS_9 Vdd N_9 N_2 Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p PD=121.8u
MNMOS_10 N_6 N_1 N_5 Gnd NMOS W=120u L=1u AS=108p PS=241.8u AD=108p PD=241.8u
MNMOS_11 N_2 N_7 Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p PD=121.8u
MNMOS_12 N_7 N_7 Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p PD=121.8u
MNMOS_1 N_8 In Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p PD=121.8u
MNMOS_2 In In Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p PD=121.8u
```

```
   MPMOS_10 N_1 N_3 Vdd Vdd PMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
   MPMOS_1 N_9 N_8 Vdd Vdd PMOS W=80u L=1u AS=72p PS=161.8u AD=72p
PD=161.8u
   MPMOS_2 N_8 N_8 Vdd Vdd PMOS W=80u L=1u AS=72p PS=161.8u AD=72p
PD=161.8u
   MPMOS_6 N_6 N_6 Vdd Vdd PMOS W=120u L=1u AS=108p PS=241.8u
AD=108p PD=241.8u
   MPMOS_7 Out N_6 Vdd Vdd PMOS W=120u L=1u AS=108p PS=241.8u
AD=108p PD=241.8u
   MPMOS_8 N_7 N_3 Vdd Vdd PMOS W=120u L=1u AS=108p PS=241.8u
AD=108p PD=241.8u
   MPMOS_9 N_3 N_3 Vdd Vdd PMOS W=120u L=1u AS=108p PS=241.8u
AD=108p PD=241.8u
   VVref1 N_5 Gnd  DC 1.5
   VVref3 N_4 Gnd  DC 1.5
   ICref1 N_3 Gnd  DC 50n
   .ends

   .subckt Current_Squaring_A In Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 N_1 N_2 Gnd NMOS W=150u L=1u AS=135p PS=301.8u
AD=135p PD=301.8u
   MNMOS_2 N_3 N_1 In Gnd NMOS W=150u L=1u AS=135p PS=301.8u AD=135p
PD=301.8u
   MNMOS_3 N_4 N_4 In Gnd NMOS W=150u L=1u AS=135p PS=301.8u AD=135p
PD=301.8u
   MNMOS_4 N_3 N_4 N_2 Gnd NMOS W=150u L=1u AS=135p PS=301.8u
AD=135p PD=301.8u
   MNMOS_5 N_2 N_2 Gnd Gnd NMOS W=150u L=1u AS=135p PS=301.8u
AD=135p PD=301.8u
   MNMOS_6 In N_2 Gnd Gnd NMOS W=150u L=1u AS=135p PS=301.8u AD=135p
PD=301.8u
   MNMOS_7 In In Gnd Gnd NMOS W=150u L=1u AS=135p PS=301.8u AD=135p
PD=301.8u
   MNMOS_8 N_2 In Gnd Gnd NMOS W=150u L=1u AS=135p PS=301.8u AD=135p
PD=301.8u
   MPMOS_1 N_5 N_5 Vdd Vdd PMOS W=300u L=2u AS=270p PS=601.8u
AD=270p PD=601.8u
   MPMOS_2 N_1 N_5 Vdd Vdd PMOS W=300u L=2u AS=270p PS=601.8u
AD=270p PD=601.8u
   MPMOS_3 N_4 N_5 Vdd Vdd PMOS W=300u L=2u AS=270p PS=601.8u
AD=270p PD=601.8u
   MPMOS_4 Out N_3 Vdd Vdd PMOS W=300u L=2u AS=270p PS=601.8u
AD=270p PD=601.8u
   MPMOS_5 N_3 N_3 Vdd Vdd PMOS W=300u L=2u AS=270p PS=601.8u
AD=270p PD=601.8u
   ICref N_5 Gnd  DC 50n
   .ends

   .subckt Current_Squaring_B In Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 N_1 N_2 Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_2 Out N_1 In Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_3 N_3 N_3 In Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_4 Out N_3 N_2 Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_5 N_2 N_2 Gnd Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_6 In N_2 Gnd Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_7 In In Gnd Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_8 N_2 In Gnd Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MPMOS_1 N_4 N_4 Vdd Vdd PMOS W=30u L=2u AS=27p PS=61.8u AD=27p
PD=61.8u
   MPMOS_2 N_1 N_4 Vdd Vdd PMOS W=30u L=2u AS=27p PS=61.8u AD=27p
PD=61.8u
   MPMOS_3 N_3 N_4 Vdd Vdd PMOS W=30u L=2u AS=27p PS=61.8u AD=27p
PD=61.8u
   ICurrentSource_1 N_4 Gnd  DC 5u
   .ends

   .subckt Current_Subtractor In1 In2 Out Vdd Vss
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 In1 In1 Vss Vss NMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   MNMOS_2 Out In1 Vss Vss NMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_1 In2 In2 Vdd Vdd PMOS W=125u L=1u AS=112.5p PS=251.8u
AD=112.5p PD=251.8u
   MPMOS_2 Out In2 Vdd Vdd PMOS W=125u L=1u AS=112.5p PS=251.8u
AD=112.5p PD=251.8u
   .ends

   .subckt INV In Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 Out In Gnd Gnd NMOS W=200u L=1u AS=180p PS=401.8u AD=180p
PD=401.8u
   MPMOS_1 Out In Vdd Vdd PMOS W=500u L=1u AS=450p PS=1.0018m
AD=450p PD=1.0018m
   .ends

   .subckt INV4 In Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 In Gnd Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
```

```
   MNMOS_2 N_2 N_1 Gnd Gnd NMOS W=20u L=1u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_3 Out N_2 Gnd Gnd NMOS W=40u L=1u AS=36p PS=81.8u AD=36p
PD=81.8u
   MPMOS_1 N_1 In Vdd Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p
PD=51.8u
   MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=50u L=1u AS=45p PS=101.8u AD=45p
PD=101.8u
   MPMOS_3 Out N_2 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   .ends

   .subckt MUX_2to1 CTRL In1 In2 Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 Out CTRL In2 Gnd NMOS W=5u L=1u AS=4.5p PS=11.8u AD=4.5p
PD=11.8u
   MNMOS_3 N_2 CTRL Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_4 N_2 In1 N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_5 N_2 Out N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_7 Out N_2 In1 Gnd NMOS W=5u L=1u AS=4.5p PS=11.8u AD=4.5p
PD=11.8u
   MNMOS_8 N_2 Out N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_9 N_2 In2 N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_1 Out N_2 In2 Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p
PD=51.8u
   MPMOS_3 N_2 CTRL Vdd Vdd PMOS W=50u L=1u AS=45p PS=101.8u AD=45p
PD=101.8u
   MPMOS_4 CTRL Out CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_5 CTRL In1 CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_7 Out CTRL In1 Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p
PD=51.8u
   MPMOS_8 CTRL In2 CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_9 CTRL Out CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   .ends

   .subckt Current_Multiplier In1 In2 Out Gnd Vdd Vss
   *-------- Devices: SPICE.ORDER == 0 --------
   XCurrent_Subtractor_1 N_4 N_6 N_1 Vdd Vss Current_Subtractor
   XCurrent_Squaring_1 N_1 Out Gnd Vdd Current_Squaring_A
   XCurrent_Squaring_B_1 N_2 Out Gnd Vdd Current_Squaring_B
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_3 In1 Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
   MNMOS_2 In1 In1 Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
   MNMOS_3 In2 In2 Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
   MNMOS_4 N_5 In2 Gnd Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
   MPMOS_1 N_3 N_3 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_2 N_2 N_3 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_3 N_4 N_3 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_4 N_2 N_5 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_5 N_5 N_5 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_6 N_6 N_5 Vdd Vdd PMOS W=100u L=1u AS=90p PS=201.8u AD=90p
PD=201.8u
   .ends

   .subckt SW_IN In Out SEL Gnd Vdd
   *-------- Devices: SPICE.ORDER == 0 --------
   XINV_1 SEL N_1 Gnd Vdd INV
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 Out SEL In Gnd NMOS W=300u L=1u AS=270p PS=601.8u AD=270p
PD=601.8u
   MPMOS_1 Out N_1 In Vdd PMOS W=350u L=1.2u AS=315p PS=701.8u
AD=315p PD=701.8u
   .ends


   ********* Simulation Settings - Parameters and SPICE Options *********

   *-------- Devices: SPICE.ORDER == 0 --------
   XCurrent_Comparator_1 N_14 N_26 N_13 Gnd Vdd Current_Comparator
   XCurrent_LPF_1 N_21 N_25 Gnd Vdd Current_LPF
   XCurrent_LPF_2 N_23 N_27 Gnd Vdd Current_LPF
   XCurrent_LPF_3 N_21 N_25 Gnd Vdd Current_LPF
   XCurrent_Subtractor_1 N_28 N_19 Out Vdd Vss Current_Subtractor
   XMUX_2to1_1 N_13 N_12 N_2 N_3 Gnd Vdd MUX_2to1
   XCurrent_Multiplier_1 N_22 N_24 N_28 Gnd Vdd Vss Current_Multiplier
   XCurrent_Multiplier_2 N_20 N_18 N_19 Gnd Vdd Vss Current_Multiplier
   XSW_IN_1 N_6 N_39 N_41 Gnd Vdd SW_IN
   XSW_IN_2 N_6 N_38 N_42 Gnd Vdd SW_IN
   XINV4_1 N_41 N_42 Gnd Vdd INV4
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 N_4 N_11 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
```

```
   MNMOS_2 N_3 N_10 Gnd Gnd NMOS W=15u L=2u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_3 N_9 N_9 N_5 Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
   MNMOS_4 N_8 N_9 N_7 Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
   MNMOS_5 Vdd N_3 N_7 Gnd NMOS W=25u L=1u AS=22.5p PS=51.8u
AD=22.5p PD=51.8u
   MNMOS_6 N_7 TEMP_CTLR Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_7 N_5 TEMP_CTLR Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_8 N_4 Vdd N_4 N_4 NMOS W=80u L=2u AS=72p PS=161.8u AD=72p
PD=161.8u
   MNMOS_9 N_3 N_4 N_3 N_3 NMOS W=2.5u L=2u AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_13 TEMP_CTLR N_30 Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_14 N_30 N_29 Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_15 N_37 Vdd Gnd Gnd NMOS W=25u L=1u AS=22.5p PS=51.8u
AD=22.5p PD=51.8u
   MNMOS_16 N_29 TEMP_CTLR N_37 Gnd NMOS W=15u L=1u AS=13.5p
PS=31.8u AD=13.5p PD=31.8u
   MNMOS_17 N_31 N_34 N_37 Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MPMOS_10 N_17 N_38 Vdd Vdd PMOS W=60u L=1u AS=54p PS=121.8u
AD=54p PD=121.8u
   MPMOS_11 N_32 N_39 Vdd Vdd PMOS W=120u L=1u AS=108p PS=241.8u
AD=108p PD=241.8u
   MPMOS_12 N_33 N_38 Vdd Vdd PMOS W=60u L=1u AS=54p PS=121.8u
AD=54p PD=121.8u
   MPMOS_13 TEMP_CTLR N_30 Vdd Vdd PMOS W=40u L=1u AS=36p PS=81.8u
AD=36p PD=81.8u
   MPMOS_14 N_30 N_29 Vdd Vdd PMOS W=40u L=1u AS=36p PS=81.8u AD=36p
PD=81.8u
   MPMOS_15 N_29 N_31 Vdd Vdd PMOS W=35u L=1u AS=31.5p PS=71.8u
AD=31.5p PD=71.8u
   MPMOS_16 N_31 N_31 Vdd Vdd PMOS W=35u L=1u AS=31.5p PS=71.8u
AD=31.5p PD=71.8u
   MPMOS_1 N_2 N_1 Vdd Vdd PMOS W=100u L=2u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_2 N_4 N_3 N_4 N_4 PMOS W=2.5u L=2u AS=2.25p PS=6.8u AD=2.25p
PD=6.8u
   MPMOS_3 Vdd N_4 Vdd Vdd PMOS W=80u L=2u AS=72p PS=161.8u AD=72p
PD=161.8u
   MPMOS_4 N_3 N_3 N_4 N_4 PMOS W=2.5u L=50u AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_5 N_1 N_1 Vdd Vdd PMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
```

```
    MPMOS_6 N_14 N_1 Vdd Vdd PMOS W=200u L=2u AS=180p PS=401.8u
AD=180p PD=401.8u
    MPMOS_7 N_12 N_1 Vdd Vdd PMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
    MPMOS_9 N_15 N_39 Vdd Vdd PMOS W=120u L=1u AS=108p PS=241.8u
AD=108p PD=241.8u
    MPMOS_18 N_6 N_6 Vdd Vdd PMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
    DDiode_1 N_34 N_16 diode AREA=1 L=0 W=0
    DDiode_2 N_16 N_35 diode AREA=1 L=0 W=0
    DDiode_3 N_35 Gnd diode AREA=1 L=0 W=0
    VVss Vss Gnd  DC 3.3
    VVdd Vdd Gnd  DC 3.3
    VVoltageSource_1 N_41 Gnd  PULSE(0 3.3 0 150m 150m 19.7 40)
    ICurrentSource_2 N_26 Gnd  DC 100n
    ICurrentSource_3 Vdd N_9  DC 100n
    ICurrentSource_4 Vdd N_34  DC 500u
    ICurrentSource_1 N_10 Gnd  PULSE(10n 100n 0 150m 150m 19.7 40)
    .PRINT TRAN V(N_41)
    VAmmeter_1 N_11 N_10 0v
    .PRINT TRAN I(VAmmeter_1)
    VAmmeter_2 N_6 N_8 0v
    .PRINT TRAN I(VAmmeter_2)
    VAmmeter_3 N_15 N_21 0v
    .PRINT TRAN I(VAmmeter_3)
    VAmmeter_4 N_17 N_23 0v
    .PRINT TRAN I(VAmmeter_4)
    VAmmeter_5 N_25 N_22 0v
    .PRINT TRAN I(VAmmeter_5)
    VAmmeter_6 N_27 N_20 0v
    .PRINT TRAN I(VAmmeter_6)
    VAmmeter_9 N_32 N_18 0v
    .PRINT TRAN I(VAmmeter_9)
    VAmmeter_10 N_33 N_24 0v
    .PRINT TRAN I(VAmmeter_10)

    ********* Simulation Settings - Analysis section *********
    .op
    .tran 0.05s 300s
    .temp 10,15,20,22,25,30,35,37,40,50,60

    ********* Simulation Settings - Additional SPICE commands *********
    .model diode D
    + is=18.8n rs=0.1 bv=2 ibv=5u cjo=30 m=0.333 n=2


    .end
```

25 resolution CMOS chip concept for the bio-inpired obstacle detection system (S-edit implementation)

```
*************** Subcircuits *****************
.subckt A0_Real InN InP Out Gnd Vdd
*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 Out N_3 N_4 N_4 NMOS W=35u L=2.5u AS=31.5p PS=71.8u AD=31.5p PD=71.8u
MNMOS_2 N_1 InN N_2 Gnd NMOS W=15u L=2.5u AS=13.5p PS=31.8u AD=13.5p PD=31.8u
MNMOS_3 N_3 InP N_2 Gnd NMOS W=15u L=2.5u AS=13.5p PS=31.8u AD=13.5p PD=31.8u
MNMOS_4 N_2 Vdd N_4 N_4 NMOS W=80u L=2.5u AS=72p PS=161.8u AD=72p PD=161.8u
MPMOS_1 Out N_3 Vdd Vdd PMOS W=75u L=2.5u AS=67.5p PS=151.8u AD=67.5p PD=151.8u
MPMOS_2 N_1 N_1 Vdd Vdd PMOS W=30u L=2.5u AS=27p PS=61.8u AD=27p PD=61.8u
MPMOS_3 N_3 N_1 Vdd Vdd PMOS W=30u L=2.5u AS=27p PS=61.8u AD=27p PD=61.8u
VVoltageSource_1 N_4 Gnd  DC -3.3
.ends

.subckt Adder In1 In2 Out Gnd Vdd Vss
*-------- Devices: SPICE.ORDER > 0 --------
MNMOS_1 N_1 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
MNMOS_2 N_1 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
MNMOS_11 N_2 In1 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
MNMOS_3 N_3 N_3 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MNMOS_12 N_7 N_7 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MNMOS_4 N_4 N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MNMOS_13 Out N_7 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MNMOS_5 N_6 N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MNMOS_14 N_2 In2 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
MNMOS_6 Out N_3 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
MPMOS_2 N_3 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u AD=76.5p PD=171.8u
```

```
   MPMOS_3  N_4  Gnd  Vdd  Vdd  PMOS  W=12u  L=48u  AS=10.8p  PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_4  N_4  Gnd  Vdd  Vdd  PMOS  W=12u  L=48u  AS=10.8p  PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_5  N_6  N_6  Vdd  Vdd  PMOS  W=85u  L=8.5u  AS=76.5p  PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_6  Out  N_6  Vdd  Vdd  PMOS  W=85u  L=8.5u  AS=76.5p  PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_11  N_7  In1  Vdd  Vdd  PMOS  W=12u  L=48u  AS=10.8p  PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_12  N_7  In2  Vdd  Vdd  PMOS  W=12u  L=48u  AS=10.8p  PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_13  Out  N_2  Vdd  Vdd  PMOS  W=85u  L=8.5u  AS=76.5p  PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_14  N_2  N_2  Vdd  Vdd  PMOS  W=85u  L=8.5u  AS=76.5p  PS=171.8u
AD=76.5p PD=171.8u
   .ends

   .subckt AMP InN InP Out Gnd
   *-------- Devices: SPICE.ORDER > 0 --------
   CCapacitor_1 N_2 Gnd  1.6u
   RResistor_1 InP InN  R=10meg
   RResistor_2 N_1 N_2  R=10k
   RResistor_3 N_3 Out  R=10
   EVCVS_E_Element_1 N_1 Gnd InP InN  100k
   EVCVS_E_Element_2 N_3 Gnd N_2 Gnd  1
   .ends

   .subckt Current_Comparator InN InP Out Gnd Vdd
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_4 N_4 Gnd Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_2 Out InN N_3 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_3 N_3 N_4 Gnd Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_4 InN InN N_4 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
   MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
   MPMOS_3 Out InP N_2 Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
   MPMOS_4 InP InP N_1 Vdd PMOS W=70u L=2u AS=63p PS=141.8u AD=63p
PD=141.8u
   .ends

   .subckt Inverter_adder In1 In2 Out Gnd Vdd Vss
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_10 N_3 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_1 N_1 In2 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_2 N_1 In1 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_3 N_2 N_2 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_4 N_4 N_4 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_5 N_6 N_4 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_6 Out N_2 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_7 Out N_8 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_8 N_8 N_8 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MNMOS_9 N_3 Gnd Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_3 N_4 In1 Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_4 N_4 In2 Vdd Vdd PMOS W=12u L=48u AS=10.8p PS=25.8u
AD=10.8p PD=25.8u
   MPMOS_5 N_6 N_6 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_6 Out N_6 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_7 Out N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   MPMOS_8 N_8 Gnd Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MPMOS_9 N_8 Gnd Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MPMOS_10 N_3 N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
   .ends

   .subckt Multiplier4cadran In1 In2 In3 In4 Out Vdd Vss
   *-------- Devices: SPICE.ORDER > 0 --------
   MNMOS_1 N_1 In1 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_2 N_1 In2 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
   MNMOS_3 N_2 N_2 Vss Vss NMOS W=80u L=8u AS=72p PS=161.8u AD=72p
PD=161.8u
```

```
    MNMOS_4 Out N_2 Vss Vss NMOS W=80u L=8u AS=72p PS=161.8u AD=72p PD=161.8u
    MNMOS_5 N_3 In3 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
    MNMOS_6 N_3 In4 Vss Vss NMOS W=3u L=60u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
    MPMOS_1 N_1 N_1 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p PD=161.8u
    MPMOS_2 N_2 N_1 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p PD=161.8u
    MPMOS_3 Out N_3 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p PD=161.8u
    MPMOS_4 N_3 N_3 Vdd Vdd PMOS W=80u L=8u AS=72p PS=161.8u AD=72p PD=161.8u
    .ends

    .subckt MUX_2to1 CTRL In1 In2 Out Gnd Vdd
    *-------- Devices: SPICE.ORDER > 0 --------
    MNMOS_1 Out CTRL In2 Gnd NMOS W=5u L=1u AS=4.5p PS=11.8u AD=4.5p PD=11.8u
    MNMOS_3 N_2 CTRL Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u AD=13.5p PD=31.8u
    MNMOS_4 N_2 In1 N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    MNMOS_5 N_2 Out N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    MNMOS_7 Out N_2 In1 Gnd NMOS W=5u L=1u AS=4.5p PS=11.8u AD=4.5p PD=11.8u
    MNMOS_8 N_2 Out N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    MNMOS_9 N_2 In2 N_2 N_2 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    MPMOS_1 Out N_2 In2 Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p PD=51.8u
    MPMOS_3 N_2 CTRL Vdd Vdd PMOS W=50u L=1u AS=45p PS=101.8u AD=45p PD=101.8u
    MPMOS_4 CTRL Out CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    MPMOS_5 CTRL In1 CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    MPMOS_7 Out CTRL In1 Vdd PMOS W=25u L=1u AS=22.5p PS=51.8u AD=22.5p PD=51.8u
    MPMOS_8 CTRL In2 CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    MPMOS_9 CTRL Out CTRL CTRL PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
    .ends

    .subckt PhotoDiode A C In Gnd
    *-------- Devices: SPICE.ORDER > 0 --------
    CCapacitor_1 N_1 A  60p
    RResistor_2 C N_1  R=10m
    DDiode_1 N_1 A diode1
    GVCCS_G_Element_1 A N_1 In Gnd  500m
    .ends

    .subckt Substractor In1 In2 Out Vdd Vss
    *-------- Devices: SPICE.ORDER > 0 --------
    MNMOS_1 N_1 N_1 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MNMOS_2 N_2 N_1 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MNMOS_3 Out N_6 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MNMOS_4 Out N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MNMOS_5 N_5 N_5 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MNMOS_6 N_4 In1 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
    MNMOS_7 N_3 In2 Vss Vss NMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
    MNMOS_8 N_6 N_6 Vss Vss NMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MPMOS_1 N_1 In1 Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
    MPMOS_2 N_2 N_2 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MPMOS_3 Out N_2 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MPMOS_4 Out N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MPMOS_5 N_5 N_4 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MPMOS_6 N_4 N_4 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MPMOS_7 N_3 N_3 Vdd Vdd PMOS W=85u L=8.5u AS=76.5p PS=171.8u
AD=76.5p PD=171.8u
    MPMOS_8 N_6 In2 Vdd Vdd PMOS W=3u L=30u AS=2.7p PS=7.8u AD=2.7p
PD=7.8u
    .ends

    .subckt Gauss_Log_Adapt_Photo In Out VBIAS Gnd Vdd
    *-------- Devices: SPICE.ORDER == 0 --------
    XMUX_2to1_1 N_13 N_11 N_12 N_1 Gnd Vdd MUX_2to1
    XPhotoDiode_1 Gnd N_10 In Gnd PhotoDiode
    XCurrent_Comparator_1 N_14 N_15 N_13 Gnd Vdd Current_Comparator
    *-------- Devices: SPICE.ORDER > 0 --------
    MNMOS_3 N_4 N_4 N_5 Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
    MNMOS_4 N_3 N_4 N_2 Gnd NMOS W=60u L=1u AS=54p PS=121.8u AD=54p
PD=121.8u
```

```
   MNMOS_5 Vdd N_1 N_2 Gnd NMOS W=25u L=1u AS=22.5p PS=51.8u
AD=22.5p PD=51.8u
   MNMOS_6 N_2 VBIAS Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_7 N_5 VBIAS Gnd Gnd NMOS W=15u L=1u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MNMOS_8 N_9 Vdd N_9 N_9 NMOS W=80u L=2u AS=72p PS=161.8u AD=72p
PD=161.8u
   MNMOS_9 N_1 N_9 N_1 N_1 NMOS W=2.5u L=2u AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MNMOS_10 Out N_7 Gnd Gnd NMOS W=30u L=1u AS=27p PS=61.8u AD=27p
PD=61.8u
   MNMOS_11 N_3 N_3 Gnd Gnd NMOS W=2u L=1u AS=1.8p PS=5.8u AD=1.8p
PD=5.8u
   MNMOS_12 N_7 N_3 Gnd Gnd NMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MNMOS_1 N_8 N_9 N_10 Gnd NMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MNMOS_2 N_1 N_10 Gnd Gnd NMOS W=15u L=2u AS=13.5p PS=31.8u
AD=13.5p PD=31.8u
   MPMOS_10 Out N_7 Vdd Vdd PMOS W=150u L=1u AS=135p PS=301.8u
AD=135p PD=301.8u
   MPMOS_11 N_3 N_3 Vdd Vdd PMOS W=10u L=1u AS=9p PS=21.8u AD=9p
PD=21.8u
   MPMOS_12 N_7 N_3 Vdd Vdd PMOS W=50u L=1u AS=45p PS=101.8u AD=45p
PD=101.8u
   MPMOS_1 N_12 N_8 Vdd Vdd PMOS W=100u L=2u AS=90p PS=201.8u AD=90p
PD=201.8u
   MPMOS_2 N_9 N_1 N_9 N_9 PMOS W=2.5u L=2u AS=2.25p PS=6.8u AD=2.25p
PD=6.8u
   MPMOS_3 Vdd N_9 Vdd Vdd PMOS W=80u L=2u AS=72p PS=161.8u AD=72p
PD=161.8u
   MPMOS_4 N_1 N_1 N_9 N_9 PMOS W=2.5u L=50u AS=2.25p PS=6.8u
AD=2.25p PD=6.8u
   MPMOS_5 N_8 N_8 Vdd Vdd PMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MPMOS_6 N_14 N_8 Vdd Vdd PMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   MPMOS_7 N_11 N_8 Vdd Vdd PMOS W=20u L=2u AS=18p PS=41.8u AD=18p
PD=41.8u
   ICurrentSource_2 N_15 Gnd  DC 100n
   ICurrentSource_3 Vdd N_4  DC 100n
   .ends

   .subckt Multiplier IN_V1 IN_V2 Out Gnd Vdd Vss
   XInverter_adder_1 IN_V1 IN_V2 N_8 Gnd Vdd Vss Inverter_adder
   XSubstractor_1 IN_V1 IN_V2 N_2 Vdd Vss Substractor
   XSubstractor_2 IN_V2 IN_V1 N_1 Vdd Vss Substractor
   XMultiplier4cadran_1 N_1 N_2 N_3 N_8 Out Vdd Vss Multiplier4cadran
   XAdder_1 IN_V1 IN_V2 N_3 Gnd Vdd Vss Adder
   .ends
```

```
    .subckt NSUM In1 In2 In3 In4 In5 In6 In7 In8 In9 In10 In11 In12 In13 In14
In15 In16
    +In17 In18 In19 In20 In21 In22 In23 In24 In25 Out Gnd Vdd
    *-------- Devices: SPICE.ORDER == 0 --------
    XA0_Real_1 N_4 N_3 N_4 Gnd Vdd A0_Real
    XA0_Real_2 N_5 N_6 Out Gnd Vdd A0_Real
    XA0_Real_3 N_1 N_2 N_3 Gnd Vdd A0_Real
    *-------- Devices: SPICE.ORDER > 0 --------
    RResistor_20 In7 N_1  R=1k
    RResistor_21 In9 N_1  R=1k
    RResistor_22 In8 N_1  R=1k
    RResistor_23 In4 N_1  R=1k
    RResistor_24 In5 N_1  R=1k
    RResistor_25 In3 N_1  R=1k
    RResistor_26 In2 N_1  R=1k
    RResistor_27 N_4 N_5  R=1k
    RResistor_28 N_5 Out  R=1k
    RResistor_29 N_6 Gnd  R=1k
    RResistor_1 In23 N_1  R=1k
    RResistor_2 In22 N_1  R=1k
    RResistor_3 In18 N_1  R=1k
    RResistor_4 In19 N_1  R=1k
    RResistor_5 In21 N_1  R=1k
    RResistor_30 In1 N_1  R=1k
    RResistor_6 In20 N_1  R=1k
    RResistor_10 In16 N_1  R=1k
    RResistor_7 N_1 N_3  R=1k
    RResistor_11 In12 N_1  R=1k
    RResistor_8 In25 N_1  R=1k
    RResistor_12 Gnd N_2  R=3k
    RResistor_9 In24 N_1  R=1k
    RResistor_13 In13 N_1  R=1k
    RResistor_14 In11 N_1  R=1k
    RResistor_15 In10 N_1  R=1k
    RResistor_16 In14 N_1  R=1k
    RResistor_17 In15 N_1  R=1k
    RResistor_18 In17 N_1  R=1k
    RResistor_19 In6 N_1  R=1k
    .ends

    .subckt NSUM_INV In1 In2 In3 In4 In5 In6 In7 In8 In9 In10 In11 In12 In13
In14 In15
    +In16 In17 In18 In19 In20 In21 In22 In23 In24 In25 Out Gnd Vdd
    *-------- Devices: SPICE.ORDER == 0 --------
    XA0_Real_3 N_2 N_5 Out Gnd Vdd A0_Real
    *-------- Devices: SPICE.ORDER > 0 --------
    RResistor_20 In7 N_2  R=1k
    RResistor_21 In9 N_2  R=1k
    RResistor_22 In8 N_2  R=1k
    RResistor_23 In4 N_2  R=1k
    RResistor_24 In5 N_2  R=1k
    RResistor_25 In3 N_2  R=1k
    RResistor_26 In2 N_2  R=1k
    RResistor_27 In1 N_2  R=1k
    RResistor_1 In23 N_2  R=1k
    RResistor_2 In22 N_2  R=1k
    RResistor_3 In18 N_2  R=1k
    RResistor_4 In19 N_2  R=1k
    RResistor_5 In21 N_2  R=1k
    RResistor_10 In16 N_2  R=1k
    RResistor_6 In20 N_2  R=1k
    RResistor_11 In12 N_2  R=1k
    RResistor_7 N_2 Out  R=1k
    RResistor_12 Gnd N_5  R=3k
    RResistor_8 In25 N_2  R=1k
    RResistor_9 In24 N_2  R=1k
    RResistor_13 In13 N_2  R=1k
    RResistor_14 In11 N_2  R=1k
    RResistor_15 In10 N_2  R=1k
    RResistor_16 In14 N_2  R=1k
    RResistor_17 In15 N_2  R=1k
    RResistor_18 In17 N_2  R=1k
    RResistor_19 In6 N_2  R=1k
    .ends

    .subckt MEMD In Out Gnd Vdd Vss
    *-------- Devices: SPICE.ORDER == 0 --------
    XSubstractor_1 N_14 N_15 Out Vdd Vss Substractor
    XMultiplier_1 N_5 N_7 N_14 Gnd Vdd Vss Multiplier
    XMultiplier_2 N_1 N_8 N_15 Gnd Vdd Vss Multiplier
    XA0_Real_1 N_5 N_6 N_5 Gnd Vdd A0_Real
    XA0_Real_2 N_6 N_16 N_6 Gnd Vdd A0_Real
    XA0_Real_3 N_7 N_1 N_7 Gnd Vdd A0_Real
    XA0_Real_4 N_8 N_5 N_8 Gnd Vdd A0_Real
    XA0_Real_6 N_2 N_3 N_1 Gnd Vdd A0_Real
    *-------- Devices: SPICE.ORDER > 0 --------
    CCapacitor_1 N_7 Gnd  700f
    CCapacitor_2 N_8 Gnd  700f
    CCapacitor_3 N_16 In  500f
    RResistor_15 Gnd N_2  R=1k
    RResistor_16 N_2 N_1  R=1k
    RResistor_17 N_3 N_6  R=500
    .ends


    ********* Simulation Settings - Parameters and SPICE Options *********

    *-------- Devices: SPICE.ORDER == 0 --------
    XMEMD_25 N_20 N_46 Gnd Vdd Vss MEMD
    XNSUM_1 N_29 N_30 N_38 N_39 N_44 N_40 N_53 N_37 N_41 N_43 N_47 N_52
N_36 N_31 N_42
```

```
    +N_49 N_50 N_34 N_33 N_45 N_48 N_51 N_35 N_32 N_46 N_3 Gnd Vdd NSUM
    XGauss_Log_Adapt_Photo_10 In7 N_5 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_11 In19 N_6 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_12 In20 N_7 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_13 In16 N_8 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_14 In15 N_9 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_15 In14 N_10 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_16 In13 N_11 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_17 In12 N_12 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_18 In11 N_13 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_19 In17 N_14 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XMEMD_1 N_4 N_29 Gnd Vdd Vss MEMD
    XMEMD_2 N_15 N_40 Gnd Vdd Vss MEMD
    XMEMD_3 N_22 N_47 Gnd Vdd Vss MEMD
    XMEMD_4 N_24 N_48 Gnd Vdd Vss MEMD
    XMEMD_5 N_23 N_49 Gnd Vdd Vss MEMD
    XMEMD_6 N_27 N_50 Gnd Vdd Vss MEMD
    XMEMD_7 N_26 N_51 Gnd Vdd Vss MEMD
    XMEMD_8 N_28 N_52 Gnd Vdd Vss MEMD
    XMEMD_9 N_5 N_53 Gnd Vdd Vss MEMD
    XGauss_Log_Adapt_Photo_20 In18 N_16 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_21 In23 N_17 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_22 In22 N_18 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_23 In21 N_19 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_24 In25 N_20 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XGauss_Log_Adapt_Photo_25 In24 N_21 VBIAS Gnd Vdd
Gauss_Log_Adapt_Photo
    XNSUM_INV_1 N_20 N_21 N_17 N_18 N_19 N_7 N_6 N_16 N_14 N_8 N_9 N_10
N_11 N_12 N_13
    +N_26 N_27 N_28 N_5 N_25 N_24 N_23 N_22 N_15 N_4 N_2 Gnd Vdd
NSUM_INV
    XAMP_1 Gnd OTA_IN- Out Gnd AMP
    XMEMD_10 N_25 N_30 Gnd Vdd Vss MEMD
    XMEMD_11 N_16 N_31 Gnd Vdd Vss MEMD
    XAdder_1 N_3 N_2 N_1 Gnd Vdd Vss Adder
    XMEMD_12 N_7 N_32 Gnd Vdd Vss MEMD
    XMEMD_13 N_6 N_33 Gnd Vdd Vss MEMD
```

```
XMEMD_14 N_10 N_34 Gnd Vdd Vss MEMD
XMEMD_15 N_9 N_35 Gnd Vdd Vss MEMD
XMEMD_16 N_11 N_36 Gnd Vdd Vss MEMD
XMEMD_17 N_12 N_37 Gnd Vdd Vss MEMD
XMEMD_18 N_13 N_38 Gnd Vdd Vss MEMD
XMEMD_19 N_8 N_39 Gnd Vdd Vss MEMD
XGauss_Log_Adapt_Photo_1 In1 N_4 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_2 In2 N_15 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_3 In3 N_22 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_4 In4 N_23 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_5 In5 N_24 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_6 In6 N_25 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_7 In10 N_26 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_8 In9 N_27 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XGauss_Log_Adapt_Photo_9 In8 N_28 VBIAS Gnd Vdd Gauss_Log_Adapt_Photo
XMEMD_20 N_14 N_41 Gnd Vdd Vss MEMD
XMEMD_21 N_17 N_42 Gnd Vdd Vss MEMD
XMEMD_22 N_18 N_43 Gnd Vdd Vss MEMD
XMEMD_23 N_19 N_44 Gnd Vdd Vss MEMD
XMEMD_24 N_21 N_45 Gnd Vdd Vss MEMD
*-------- Devices: SPICE.ORDER > 0 --------
CCapacitor_1 OTA_IN- Out  1p
RResistor_1 N_1 OTA_IN-  R=10k

********* Simulation Settings - Analysis section *********

********* Simulation Settings - Additional SPICE commands *********

.end
```

Java code implementation in Android SDK using OpenCV for the bio-inspired obstacle detection system

```
package org.opencv.samples.imagemanipulations;

import org.opencv.android.Utils;
import org.opencv.core.Core;
import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.core.Point;
import org.opencv.core.Scalar;
import org.opencv.core.Size;
import org.opencv.highgui.Highgui;
import org.opencv.highgui.VideoCapture;
import org.opencv.imgproc.Imgproc;

import android.content.Context;
import android.graphics.Bitmap;
import android.util.Log;
import android.view.SurfaceHolder;

class ImageManipulationsView extends SampleCvViewBase {
    private Mat mRgba;
    private Mat mGray;
    private Mat mIntermediateMat;
    //private static Mat aux;

    private static int n0, z4, n4, z0 ,s;

    private Mat mRgbaInnerWindow;
    private Mat mGrayBigWindow;
    private Mat mGraySmallWindow;
    private Mat mBlurWindow;
    private Mat mZoomWindow;
    private Mat mZoomCorner;
    private Mat mArea;
    private static Mat frame0,frame1,frame2,frame3,frame4;
    private static Mat zoom0,zoom1,zoom2,zoom3,zoom4;
    private static int index;

    private Mat mSepiaKernel;

    private boolean ok_sound, ok_sound2;


    public ImageManipulationsView(Context context) {
        super(context);
        ok_sound = ok_sound2 = true;
        frame0 = new Mat();
        frame1 = new Mat();
        frame2 = new Mat();
        frame3 = new Mat();
        frame4 = new Mat();
        zoom0 = new Mat();
        zoom1 = new Mat();
        zoom2 = new Mat();
        zoom3 = new Mat();
        zoom4 = new Mat();

        index = 4;
        n0 = z4 = n4 = z0 = s = 0;
        mSepiaKernel = new Mat(4, 4, CvType.CV_32F);
        mSepiaKernel.put(0, 0, /* R */0.189f, 0.769f, 0.393f, 0f);
        mSepiaKernel.put(1, 0, /* G */0.168f, 0.686f, 0.349f, 0f);
        mSepiaKernel.put(2, 0, /* B */0.131f, 0.534f, 0.272f, 0f);
        mSepiaKernel.put(3, 0, /* A */0.000f, 0.000f, 0.000f, 1f);
    }

    @Override
```

```
        public void surfaceChanged(SurfaceHolder _holder, int format, int width, int
height) {
            super.surfaceChanged(_holder, format, width, height);

            synchronized (this) {
                // initialize Mats before usage
                mGray = new Mat();
                mRgba = new Mat();
                mIntermediateMat = new Mat();
            }
        }

        private void CreateAuxiliaryMats() {
            if (mRgba.empty())
                return;

            int rows = mRgba.rows();
            int cols = mRgba.cols();

            if (mRgbaInnerWindow == null)
                mRgbaInnerWindow = mRgba.submat(rows/2-50, rows/2+50, cols/2-
50, cols/2+50);

            if (mGrayBigWindow == null && !mGray.empty())
                      mGrayBigWindow = mGray.submat(rows/2-50, rows/2+50, cols/2-
50, cols/2+50);



          /*
           Bitmap      bmp2      =      Bitmap.createBitmap(mGrayInnerWindow.cols(),
mGrayInnerWindow.rows(), Bitmap.Config.ARGB_8888);

            Utils.matToBitmap(mGrayInnerWindow, bmp1);

            int      width1 = bmp1.getWidth();
           int       height1 = bmp1.getHeight();

           for(int x=20;x<width1;x++)
                        for(int y=20;y<height1;y++) {
                                   bmp1.setPixel(x, y, Color.WHITE);

                        }

           mGrayInnerWindow =Utils.bitmapToMat(bmp1); */

            if (mBlurWindow == null)
                mBlurWindow = mRgba.submat(0, rows, cols / 3, cols * 2 / 3);

            if (mZoomCorner == null)
```

```
                mZoomCorner = mRgba.submat(0, rows / 2 - rows / 10, 0, cols / 2 -
cols / 10);

            if (mZoomWindow == null)
                mZoomWindow = mRgba.submat(rows / 2 - 9 * rows / 100, rows / 2 +
9 * rows / 100, cols / 2 - 9 * cols / 100, cols / 2 + 9 * cols / 100);

            if (mArea == null)
                    mArea = mRgba.submat(0, rows / 2 - rows / 10, 0, cols / 2 - cols /
10);

        }

        public static int summPixels(Mat m)
        {
          int s=0;
          for (int i=0; i<m.width(); i++)
                  for (int j=0; j<m.height(); j++)
                  {
                          if( m.get(j, i)[0] > 1 )
                                  s+=1;
                  }
          return s;
        }

        public int doIt(Mat m, Mat z)
        {
          if (index == 4)
          {
                  frame4 = m.clone();
                  zoom4 = z.clone();
          }
          if (index == 3)
          {
                  frame3 = m.clone();
                  zoom3 = z.clone();
          }
          if (index == 2)
          {
                  frame2 = m.clone();
                  zoom2 = z.clone();
          }
          if (index == 1)
          {
                  frame1 = m.clone();
                  zoom1 = z.clone();
          }
          if (index == 0)
          {
                  frame0 = m.clone();
                  zoom0 = z.clone();
          }


          if (index<=0)
          {
                  s = 0;
                          for (int i=0; i<100; i++)
                                  for (int j=0; j<100; j++)
                                  {
                                          n0 = (frame0.get(j, i)[0]>1) ? 1 : 0;
                                          n4 = (frame4.get(j, i)[0]>1) ? 1 : 0;
                                          z0 = (zoom0.get(j, i)[0]>1) ? 1 : 0;
                                          z4 = (zoom4.get(j, i)[0]>1) ? 1 : 0;
                                          s += (n0*(1-z4)-n4*z0);
                                  }

                          frame1=frame0.clone();
                          frame2=frame1.clone();
                          frame3=frame2.clone();
                          frame4=frame3.clone();
                          zoom1=zoom0.clone();
                          zoom2=zoom1.clone();
                          zoom3=zoom2.clone();
                          zoom4=zoom3.clone();


                          return s;
          }
          else index--;

          return 0;
        }
      //Core.rectangle(mGrayBigWindow, new Point(mGrayBigWindow.width()/2-
50,mGrayBigWindow.height()/2-50), new
Point(mGrayBigWindow.width()/2+50,mGrayBigWindow.height()/2+50), new
Scalar(255, 0, 0, 255), 7);
        public void playSound()
        {
        if (s>800)
        {
        if (ImageManipulationsActivity.loaded && ok_sound) {

ImageManipulationsActivity.soundPool.play(ImageManipulationsActivity.sound, 7, 7,
1, 0, 1f);
        ok_sound = false;
        ok_sound2 = true;
        }
        }else
        if (s>600)
        {
        if (ImageManipulationsActivity.loaded && ok_sound2) {
```

```
ImageManipulationsActivity.soundPool.play(ImageManipulationsActivity.sound2,   7,
7, 1, 0, 1f);
        ok_sound2 = false;
        ok_sound = true;
        }
        } else
        {
        ok_sound = ok_sound2 = true;
        }


        }


        @Override
        protected Bitmap processFrame(VideoCapture capture) {
            switch (ImageManipulationsActivity.viewMode) {

            case ImageManipulationsActivity.VIEW_MODE_RGBA:
                capture.retrieve(mRgba,
Highgui.CV_CAP_ANDROID_COLOR_FRAME_RGBA);
                break;

            case ImageManipulationsActivity.VIEW_MODE_CANNY:
                capture.retrieve(mRgba,
Highgui.CV_CAP_ANDROID_COLOR_FRAME_RGBA);
                capture.retrieve(mGray, Highgui.CV_CAP_ANDROID_GREY_FRAME);

       /*          Bitmap  bmp1  =  Bitmap.createBitmap(mGray.cols(),  mGray.rows(),
Bitmap.Config.ALPHA_8);
                int  width1 = bmp1.getWidth();
                      int       height1 = bmp1.getHeight();


               Color w =new Color();


                      for(int x=0;x<width1;x++)
                        for(int y=0;y<height1;y++) {
                                   bmp1.setPixel(x, y, Color.WHITE);

                        }

                      mGrayInnerWindow=Utils.bitmapToMat(bmp1);
            */
                      //capture.retrieve(mGrayInnerWindow,
Highgui.CV_CAP_ANDROID_GREY_FRAME);

                if (mRgbaInnerWindow == null || mGrayBigWindow == null)
                    CreateAuxiliaryMats();
```

```
             // System.out.print("Rezultat e " );



              Imgproc.Canny(mGrayBigWindow, mGrayBigWindow, 80, 90);

              //mGrayBigWindow                                          =
mGrayBigWindow.submat(mGrayBigWindow.height()/2-50,
mGrayBigWindow.height()/2+50,                  mGrayBigWindow.width()/2-50,
mGrayBigWindow.width()/2+50);

              mGraySmallWindow                                        =
mGrayBigWindow.submat(mGrayBigWindow.height()/4,mGrayBigWindow.height()*
3/4, mGrayBigWindow.width()/4,mGrayBigWindow.width()*3/4);

              Imgproc.resize(mGraySmallWindow,              mGraySmallWindow,
mGrayBigWindow.size());

             doIt(mGrayBigWindow, mGraySmallWindow);

              if (s!=0)
              {
                   //Log.d("SUM",""+s);

                   ////// treshold = 400
                            playSound();

              }




             Imgproc.cvtColor(mGrayBigWindow,                mRgbaInnerWindow,
Imgproc.COLOR_GRAY2BGRA, 4);
            // zoom(capture);
              break;

           case ImageManipulationsActivity.VIEW_MODE_SOBEL:
              capture.retrieve(mRgba,
Highgui.CV_CAP_ANDROID_COLOR_FRAME_RGBA);
              capture.retrieve(mGray, Highgui.CV_CAP_ANDROID_GREY_FRAME);

              if (mRgbaInnerWindow == null || mGrayBigWindow == null)
                 CreateAuxiliaryMats();

              Imgproc.Sobel(mGrayBigWindow, mIntermediateMat, CvType.CV_8U, 1,
1);
              Core.convertScaleAbs(mIntermediateMat, mIntermediateMat, 10);
              Imgproc.cvtColor(mIntermediateMat,              mRgbaInnerWindow,
Imgproc.COLOR_GRAY2BGRA, 4);
              break;
            case ImageManipulationsActivity.VIEW_MODE_SEPIA:
                capture.retrieve(mRgba,
Highgui.CV_CAP_ANDROID_COLOR_FRAME_RGBA);
                Core.transform(mRgba, mRgba, mSepiaKernel);
                break;

            case ImageManipulationsActivity.VIEW_MODE_BLUR:
                capture.retrieve(mRgba,
Highgui.CV_CAP_ANDROID_COLOR_FRAME_RGBA);
                if (mBlurWindow == null)
                    CreateAuxiliaryMats();
                Imgproc.blur(mBlurWindow, mBlurWindow, new Size(15, 15));
                break;

            case ImageManipulationsActivity.VIEW_MODE_ZOOM:
                      zoom(capture);
                      break;
                    }

            Bitmap bmp = Bitmap.createBitmap(mRgba.cols(), mRgba.rows(),
Bitmap.Config.ARGB_8888);

            if (Utils.matToBitmap(mRgba, bmp))
                return bmp;

            bmp.recycle();
            return null;
        }
        public void zoom(VideoCapture capture)
        {

            capture.retrieve(mRgba,
Highgui.CV_CAP_ANDROID_COLOR_FRAME_RGBA);
            if (mZoomCorner == null || mZoomWindow == null)
                CreateAuxiliaryMats();
            Imgproc.resize(mZoomWindow, mZoomCorner, mZoomCorner.size());

            Size wsize = mZoomWindow.size();
            Core.rectangle(mZoomWindow, new Point(1, 1), new Point(wsize.width - 2,
wsize.height - 2), new Scalar(255, 0, 0, 255), 2);


        }

        @Override
        public void run() {
            super.run();

            synchronized (this) {
                // Explicitly deallocate Mats
                if (mZoomWindow != null)
                    mZoomWindow.release();
                if (mZoomCorner != null)
                    mZoomCorner.release();
                if (mBlurWindow != null)
                    mBlurWindow.release();
                if (mGrayBigWindow != null)
                    mGrayBigWindow.release();
                if (mGraySmallWindow != null)
                    mGraySmallWindow.release();
                if (mRgbaInnerWindow != null)
                    mRgbaInnerWindow.release();
                if (mRgba != null)
                    mRgba.release();
                if (mGray != null)
                    mGray.release();
                if (mIntermediateMat != null)
                    mIntermediateMat.release();

                mRgba = null;
                mGray = null;
                mIntermediateMat = null;
                mRgbaInnerWindow = null;
                mGrayBigWindow = null;
                mGraySmallWindow = null;
                mBlurWindow = null;
                mZoomCorner = null;
                mZoomWindow = null;
            }
        }
    }
```

package org.opencv.samples.imagemanipulations;

```
import android.app.Activity;
import android.media.AudioManager;
import android.media.SoundPool;
import android.media.SoundPool.OnLoadCompleteListener;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.Window;

public class ImageManipulationsActivity extends Activity {
       public  static  SoundPool soundPool;
    public static boolean loaded;

public static int sound;
public static int sound2;
```

```
  private static final String TAG           = "Sample::Activity";

public static final int    VIEW_MODE_RGBA  = 0;
public static final int    VIEW_MODE_CANNY = 1;
public static final int    VIEW_MODE_SEPIA = 2;
public static final int    VIEW_MODE_SOBEL = 3;
public static final int    VIEW_MODE_BLUR  = 4;
public static final int    VIEW_MODE_ZOOM  = 5;

private MenuItem          mItemPreviewRGBA;
private MenuItem          mItemPreviewCanny;
private MenuItem          mItemPreviewSepia;
private MenuItem          mItemPreviewSobel;
private MenuItem          mItemPreviewBlur;
private MenuItem          mItemPreviewZoom;

public static int          viewMode        = VIEW_MODE_RGBA;

public ImageManipulationsActivity() {
   Log.i(TAG, "Instantiated new " + this.getClass());
}

/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {
   Log.i(TAG, "onCreate");
   super.onCreate(savedInstanceState);
   requestWindowFeature(Window.FEATURE_NO_TITLE);
   setContentView(new ImageManipulationsView(this));
 // Set the hardware buttons to control the music
   this.setVolumeControlStream(AudioManager.STREAM_MUSIC);
   // Load the sound
   soundPool = new SoundPool(10, AudioManager.STREAM_MUSIC, 0);
   soundPool.setOnLoadCompleteListener(new OnLoadCompleteListener() {

   public void onLoadComplete(SoundPool soundPool, int sampleId,
   int status) {
   loaded = true;
   }
   });
   sound = soundPool.load(this, R.raw.blip, 1);
   sound2 = soundPool.load(this, R.raw.blip2, 1);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
   Log.i(TAG, "onCreateOptionsMenu");
   mItemPreviewRGBA = menu.add("Preview RGBA");
   mItemPreviewCanny = menu.add("Canny");
   mItemPreviewSepia = menu.add("Sepia");
   mItemPreviewSobel = menu.add("Sobel");
```

```
        mItemPreviewBlur = menu.add("Blur");
        mItemPreviewZoom = menu.add("Zoom");
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        Log.i(TAG, "Menu Item selected " + item);
        if (item == mItemPreviewRGBA)
            viewMode = VIEW_MODE_RGBA;
        else if (item == mItemPreviewCanny)
            viewMode = VIEW_MODE_CANNY;
        else if (item == mItemPreviewSepia)
            viewMode = VIEW_MODE_SEPIA;
        else if (item == mItemPreviewSobel)
            viewMode = VIEW_MODE_SOBEL;
        else if (item == mItemPreviewBlur)
            viewMode = VIEW_MODE_BLUR;
        else if (item == mItemPreviewZoom)
            viewMode = VIEW_MODE_ZOOM;
        return true;
    }
}


package org.opencv.samples.imagemanipulations;

import java.util.List;

import org.opencv.core.Size;
import org.opencv.highgui.VideoCapture;
import org.opencv.highgui.Highgui;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.util.Log;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

public abstract class SampleCvViewBase extends SurfaceView implements
SurfaceHolder.Callback, Runnable {
    private static final String TAG = "Sample::SurfaceView";

    private SurfaceHolder       mHolder;
    private VideoCapture        mCamera;
    private FpsMeter           mFps;

    public SampleCvViewBase(Context context) {
        super(context);
        mHolder = getHolder();
```

```
      mHolder.addCallback(this);
      mFps = new FpsMeter();
      Log.i(TAG, "Instantiated new " + this.getClass());
    }

    public void surfaceChanged(SurfaceHolder _holder, int format, int width, int
height) {
      Log.i(TAG, "surfaceCreated");
      synchronized (this) {
        if (mCamera != null && mCamera.isOpened()) {
          Log.i(TAG, "before mCamera.getSupportedPreviewSizes()");
          List<Size> sizes = mCamera.getSupportedPreviewSizes();
          Log.i(TAG, "after mCamera.getSupportedPreviewSizes()");
          int mFrameWidth = width;
          int mFrameHeight = height;

          // selecting optimal camera preview size
          {
            double minDiff = Double.MAX_VALUE;
            for (Size size : sizes) {
              if (Math.abs(size.height - height) < minDiff) {
                mFrameWidth = (int) size.width;
                mFrameHeight = (int) size.height;
                minDiff = Math.abs(size.height - height);
              }
            }
          }

          mCamera.set(Highgui.CV_CAP_PROP_FRAME_WIDTH,
mFrameWidth);
          mCamera.set(Highgui.CV_CAP_PROP_FRAME_HEIGHT,
mFrameHeight);
        }
      }
    }

    public void surfaceCreated(SurfaceHolder holder) {
      Log.i(TAG, "surfaceCreated");
      mCamera = new VideoCapture(Highgui.CV_CAP_ANDROID);
      if (mCamera.isOpened()) {
        (new Thread(this)).start();
      } else {
        mCamera.release();
        mCamera = null;
        Log.e(TAG, "Failed to open native camera");
      }
    }

    public void surfaceDestroyed(SurfaceHolder holder) {
      Log.i(TAG, "surfaceDestroyed");
      if (mCamera != null) {
```

```
                synchronized (this) {
                    mCamera.release();
                    mCamera = null;
                }
            }
        }

        protected abstract Bitmap processFrame(VideoCapture capture);

        public void run() {
            Log.i(TAG, "Starting processing thread");
            mFps.init();

            while (true) {
                Bitmap bmp = null;

                synchronized (this) {
                    if (mCamera == null)
                        break;

                    if (!mCamera.grab()) {
                        Log.e(TAG, "mCamera.grab() failed");
                        break;
                    }

                    bmp = processFrame(mCamera);


                    mFps.measure();
                }

                if (bmp != null) {
                    Canvas canvas = mHolder.lockCanvas();
                    if (canvas != null) {
                        canvas.drawBitmap(bmp, (canvas.getWidth() - bmp.getWidth()) /
2, (canvas.getHeight() - bmp.getHeight()) / 2, null);
                        mFps.draw(canvas, (canvas.getWidth() - bmp.getWidth()) / 2, 0);
                        mHolder.unlockCanvasAndPost(canvas);
                    }
                    bmp.recycle();
                }
            }

            Log.i(TAG, "Finishing processing thread");
        }
    }
```