The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania

DOCTORAL THESIS
in the field of
Robotics

Computational Sensors for Global Operations in Vision

Vladimir M. Brajovic

January 22, 1996

Submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy.

Computational Sensors for Global Operations in Vision

Vladimir Brajovic

January 30, 1996

CMU–RI–TR–96–02

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.

This research was partially supported by Office of Naval Research (ONR) under contract N00014–95–1–0591 and National Scientific Foundation (NSF) under contract MIP–9305494. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ONR, NSF or the U.S. government.

ACM Computing Reviews Keywords: Computational Sensors, Smart Sensors, VLSI, Computer Vision, Image Processing.

Carnegie Mellon

Robotics

Thesis

Computational Sensors for Global Operations in Vision

Vladimir M. Brajovic

Submitted in Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in the field of Robotics

ACCEPTED:

Takeo Kanade — Thesis Committee Chair	March 28, 1996 — Date
Matthew T. Mason — Program Chair	March 27, 1996 — Date
Raj Reddy — Dean	4/21/1996 — Date

APPROVED:

Paul Christiano — Provost	25 April 1996 — Date

To Milanka, Miodrag
and
Denise.

Abstract

The performance of existing machine vision systems still significantly lags that of a biological vision. The two most critical features presently missing from the machine vision are *low latency processing* and *top–down sensory adaptation*. This thesis proposes to overcome these two deficiencies by implementing *global operations in computational sensors*.

Computational sensors incorporate computation at the level of sensing and can both reduce latency and facilitate top–down sensory adaptation. In the context of this thesis the global operations are important because: (1) in perception each decision is a kind of global, or overall, conclusion necessary for the coherent interaction with the environment, and (2) global operations produce *a few* quantities for the description of the environment which can be *quickly* transferred and/or processed to produce an appropriate action for a machine.

The main difficulty with implementing global operations comes from the necessity to bring together all or most of the data in the input data set. This work proposes two mechanisms for implementing global operations in computational sensors: (1) *sensory attention*, and (2) *intensity–to–time processing paradigm*.

The sensory attention is based on the premise that salient features within the retinal image represent important global features of the entire image. By

selecting a small region of interest around the salient feature for subsequent processing, the sensory attention eliminates extraneous information and allows the processor to handle small amount of data at a time. The sensory attention is used for a VLSI implementation of a *tracking computational sensor* — a computational sensor that attends and tracks a visual stimuli in the retinal image.

The *intensity–to–time processing paradigm* is based on the notion that stronger signals elicit responses before weaker ones allowing a global processor to make decisions based only on a few inputs at a time. The key is that some preliminary decisions about the retinal image can be made as soon as the first responses are received. The intensity–to–time processing paradigm is used for the VLSI implementation of a *sorting computational sensor* — a computational sensor that sorts input stimuli by their intensity as they are being sensed.

By implementing the tracking and sorting sensors it is demonstrated that the computational sensor paradigm improves latency and provides top–down sensory feedback for more robust performance in computer vision systems.

Acknowledgments

I wish to thank my mentor Dr. Takeo Kanade for his support and technical advice. As a scientist and leader of a great vision, Takeo has not only provided invaluable feedback, but has given me the latitude I have needed to develop as a researcher. Also, I would like to thank my thesis committee members Dr. Rick Carley, Dr. Steve Shafer and Dr. Andreas Andreou for their insightful analysis and valuable comments concerning my work.

I owe much to all the members of The Robotics Institute. Their technical excellence and companionship stimulated most of my ideas and made my work both possible and enjoyable. I would like to specifically thank Bill Ross, Omead Amidi and Amy Roch for their help.

I would like to thank my parents, Milanka and Miodrag, for continuously encouraging my pursuits in life, and for all the sacrifices they have made to provide me with a world of opportunity. Finally, I am especially grateful to my wife, Denise Susan, for her love, patience, encouragement and comfort.

Vladimir Brajovic

January 22, 1996.

Contents

Chapter 1

Introduction

Perception plays a dominant role in the development of an intelligent behavior. Our awareness of the environment relies on the activity of our sense organs. These outposts of the nervous system translate environmental changes into activity in sensory nerve fibers. It is then the function of the central nervous system to interpret this sensory information, integrating it into an appropriate pattern of behavior. Like biological systems, intelligent robotic behavior relies heavily on the sensory perception. Especially rich in information, and fascinating in its capability, is *vision*. It is not surprising that vision research has received equally high interest in neurophysiology, psychology, computer science and engineering.

In the last 30 years machine vision research advanced along many fronts. Cameras have improved: their resolution and sensitivity have increased, and new sensors such as uncooled infrared cameras are now commercially available. Many recognition algorithms have been developed: from 3D model matching to artificial neural networks. Yet performance of the existing machine vision systems still significantly lags that of biological vision.

Most notably, the machine vision is limited in its ability to *quickly* and *reliably* notice changes in the environment.

This thesis is about using VLSI technology to improve the visual sensing and processing for faster and more reliable performance. Its contribution is towards a more capable, affordable, smaller and low–power machine vision task–oriented component which will make many new practical applications possible.

1.1 Motivation

The fundamental problem in machine vision comes from the computational complexity of basic tasks. Examples include the problem of detecting a target element in an image (*visual search*) and the problem of finding a correspondence between the image and a set of models (*matching*). Any algorithm which solves these problems in a general way, without the help of assumptions and heuristics, requires exponential execution time as a function of the image size and the number of stored models. From this observation it becomes apparent that vision systems have limited capability to scale up with images of increasing size and complexity.

The consistent paradigm in machine vision has been that a "camera" sees the world and a computer "algorithm" recognizes the object. Implicit in this view is the separation between the camera — a sensing device for transducing spatio–spectral–temporal phenomena to electric signals, and the computer — a computational device for processing and making sense out of data. That is, the transduced signal is read out of the sensor and digitized into the computer for processing. The separation of sensing and processing has resulted in several deficiencies in the computer vision systems developed so far. The two most critical features missing from the sequential paradigm are *low latency processing* and *top–down sensory adaptation*.

Latency, or reaction time, is the time that a system takes to react to an event. For example, a standard video camera takes 1/30 of a second to transfer an image. In many robotic applications it is too late by the time the system receives the image from such a camera. As another example, pipelined dedicated vision hardware can deliver the processing power to update its output 30 times per second, but the latency incurred through the pipeline is typically several seconds. These examples point to two primary sources of latency in vision systems: *the data transfer bottleneck* caused by the need to transfer an image from the camera to the processor, and *the computational*

load bottleneck caused by the processor's inability to quickly handle the large amount of data. The detrimental effects of both bottlenecks scale–up with the image size.

Another aspect that has been neglected in machine vision is top–down sensory adaptation. Many learning algorithms have been developed that adjust to variations in appearance of an object in sensor images. Nevertheless, complex ad–hoc algorithms that try to extract relevant information from inadequate sensor data are inevitably unreliable. In fact, time and time again it has been observed that using the most appropriate sensing modality or setup, allows recognition algorithms to be far simpler and more reliable. For example, the concept of active vision proposes to control the geometric parameters of the camera (e.g. pan, tilt, etc.) to improve the reliability of the perception [4]. It has been shown that initially ill–posed problems can be solved after the top–down adaptation of the camera's pose has acquired new more appropriate image data. However, adjusting geometric parameters is only one level at which adaptation can take place. A system that can adjust its operation at all levels, even down to the point of sensing, would be far more adaptive than the one that tries to cope with the variations at the "algorithmic" or "motoric" level alone.

The lack of fast processing and top–down sensory adaptation in the sense–then–process paradigm, suggest that an alternative is needed.

Compared to the capabilities of the available machine vision systems and techniques, the performance of biological vision is astonishing. It has been estimated that humans can recognize up to 100,000 objects within 100–200 ms [54]. In addition, the recognition has a high degree of invariance with respect to factors such as the position, scale and orientation, which may completely change the retinal image of objects.

One of the most important factors which determines these capabilities is the high number of processing elements (approx. 10^{11} neurons) working in parallel in the human brain. However, given the relatively slow response of each neuron and the huge amount of input data (approx. 10^8 receptors), it becomes apparent that the sheer number of neurons is not sufficient to explain these performances. In fact, the human visual system is not even structured to exploit the computational power of a single, fully–connected network of cells; it is rather organized into a number of areas analyzing different aspects of the image.

At the very first stage of the processing hierarchy is the retina [19]. The retina senses visual information and transmits it to the brain via the optical nerve (approx. 1.5 x 10^6 fibers). While the number of fibers in the optical nerve is far beyond what we can replicate in an artificial system at the moment, it is far below the number of photoreceptors in the eye (approx. 10^8 receptors). If we further consider that some fibers respond only to motion and other transmit contrast rather than the photometric information of the receptors, it becomes obvious that this fascinating layer of neural tissue carries out some form of processing and data reduction. Indeed, the optical nerve fibers are axons derived from fourth or fifth order neurons in the visual pathway [32], i.e. there are four or five layers of neurons processing receptors signals before the information is sent through the optical nerve.

The eye processes optical information even before the light is transduced into the neural signals; in addition to the lens focuses and the iris for rudimentary intensity adaptation, the photosensitive elements of the retina are spatially organized in a non–uniform way . The high spatial resolution of the fovea allows detailed sensing in the central region, while keeping a vague representation of the periphery of the image. The drawback of this strategy is the need for eye movement, which sequentially shifts the fovea to the "interesting" parts of the image.

In addition to these anatomical mechanisms for information compression, functional mechanisms exist in the higher processing centers of the brain. An example is attention — the ability to select a part of the retinal image to which the application of higher level processes can be restricted[3][20][54]. Unlike eye movement, the attention shifts do not require any motor action, but occur internally, on a fixed retinal image. For this reason, attention shifts are faster than eye movements and appear to rapidly determine a number of interesting locations of the image. Then, the top–down pathways may initiate the eye movements for foveating onto one of these locations.

From this discussion it becomes evident that biological vision tightly couples sensing with processing and provides the top–down feedback for sensory adaptation and eye movement.

1.2 Computational Sensor Paradigm

Computational sensors [37] mimic one aspect of biological systems: they incorporate computation at the level of sensing to improve performance and achieve new capabilities which were not otherwise possible. Computational

sensors are usually VLSI circuits which may (1) include on–chip processing elements tightly coupled with on–chip sensors, (2) exploit unique optical design or geometrical arrangement of elements, or (3) use the physics of the underlying material for computation.

The computational sensor paradigm has potential to both reduce latency and facilitate top–down sensory adaptation, two main deficiencies of the computer vision at the moment. Namely, by integrating sensing and processing on a VLSI chip both transfer and computational bottlenecks can be alleviated: on–chip routing provides high capacity transfer, while an on-chip processor may implement massively–parallel fine–grain computation providing high processing capacity which readily scales up with the image size. In addition, the tight coupling between processor and sensor provides opportunity for a fast processor–sensor feedback for top–down adaptation.

1.3 Global vs. Local Operations

Integrating sensing and processing is not a new idea. So far, however, a great majority of the solutions focused on image processing. There are numerous computational sensors which implement *local operations* on a single light sensitive VLSI chip [7] [30] [38] [47] [49] [68] [74]. The local operations use operands within a small spatial/temporal neighborhood of data and land themselves to the graceful implementation in VLSI. A typical example is the invariant finite impulse response (FIR) filtering, such as smoothing or edge detection. While computationally demanding, the local operations produce preprocessed images; therefore, a large quantity of data still must be read out and further inspected before a decision for an appropriate action is made. Consequently, a great majority of computational sensors built thus far are limited in their ability to quickly respond to changes in the environment.

On the other hand, *global operations* result in fewer entities for description of a scene. If computed at the point of sensing, these entities could be routed from a computational sensor through a few output pins without causing the transfer bottleneck. This information will be often sufficient for rapid decision making and the actual image need not be read out. Global operations, however, need to gather and process information over the entire set of data. This global exchange of data among a large number of processors/sites quickly saturates communication connections and adversely affects computing efficiency in parallel systems — parallel digital computers and

computational sensors alike. It is not surprising that there are only a few computational sensors for global signal aggregation: sensors for computing position of a bright region on a dark background [18] [66], and a sensor for motion estimation over the entire retinal image [67].

1.4 Preview of the Main Result

The primary goal of this thesis is to design computational sensors which reduce the latency in a vision system, and provide top–down feedback for more reliable performance. Such computational sensors must quickly provide reliable information for appropriate action for a task at hand. To attain this goal this work embarks upon the problem or implementing global operations in computational sensors.

The main problem with global operations comes from the necessity to bring together, or aggregate all (or most) of the data in the input data set. There are fundamental differences in how biological and artificial systems aggregate input signals. In digital systems, for example, gates with fan–in greater than 4 are rarely employed. The fan–in of an average neuron is 1,000 to 3,000, or even 10,000 [53]. Each input requires that a signal is routed to it. The more input signals, the more wiring is required, in both biological and artificial systems. Wires do not process information; therefore, economizing on wire should be important priority for both nerves and chips. Yet, the biological systems opted for large fan-ins. Some researchers [53] hypothesize that each neuron must have synaptic inputs representing *all* features that might ever be used, even though only a subset of them will contribute to any particular decision. Thus, it seems that the neurons are optimized for making global decisions about a large number of inputs, but using only a few of those inputs at a time.

This work is concerned with efficient implementation of global operations over a large groups of image data using computational sensor paradigm. In order to overcome obvious technological limitation for quickly communicating and processing large amounts of data, the proposed solutions draw upon the experiences of evolution and suggests the following implementation. The data are supplied optically by focusing data (henceforth referred to as a retinal image) onto the array of photodetectors. A processor integrated within the chip, mimics neurons and makes a decision *based only on a few input data at a time*. The problem is how to efficiently chose which few input data to route to the global processor at each given time. Two

models are investigated: *the sensory attention*, and *the intensity–to–time processing paradigm.*

Sensory Attention

The *sensory attention* follows the model of *visual attention* in brains. This analogy is attractive for two reasons. First, the main argument that has been used to explain the need for selective visual attention in brains is that there exist some kind of processing and communication limitation in the visual system. So it does in machines. Attention "funnels" only relevant information and protect the limited communication and processing resources from the information overload. Second, it has been shown that the visual attention improves performance, and is needed for maintaining coherent behavior while interacting with the environment (i.e. attention–for–action) [3]. Location of such attention must be maintained in the environmental coordinates; thus maintaining coherence under ocular and head motion [54]. The attention–for–action model is consistent with the goals of producing reliable fast–reacting computational sensors.

For implementation of attention several problems must be solved: (1) how to select interesting location within the retinal image, (2) how to shift the attention to another location, and (3) how to transfer data from focus of attention to the central processor for further inspection.

A traditional model for implementing visual attention is shown in Figure 1a. The winner–take–all (WTA) has been suggested for implementing location selection [43] [42]. The WTA determines the identity and magnitude of its strongest input [23]. The WTA uses a saliency map to guide the attention to the most conspicuous part of the retinal image. The saliency map can be derived from image features including the intensity, color, spatial and temporal derivatives, motion, and orientation.

In the prototype implementation of the sensory attention proposed by this work, our concern is not how to compute the saliency map, but rather how to quickly and reliably locate and maintain interesting location in the saliency map. At the present state of technology we deliver the saliency map optically by focusing it onto the array of photodetectors feeding the WTA network (Figure 1b). This embodiment of the sensory attention we call *tracking computational sensor* because when the saliency map is a natural image, the trivial saliency map, the features that attract attention are bright spots in the environment. The sensor selects and tracks those spots.

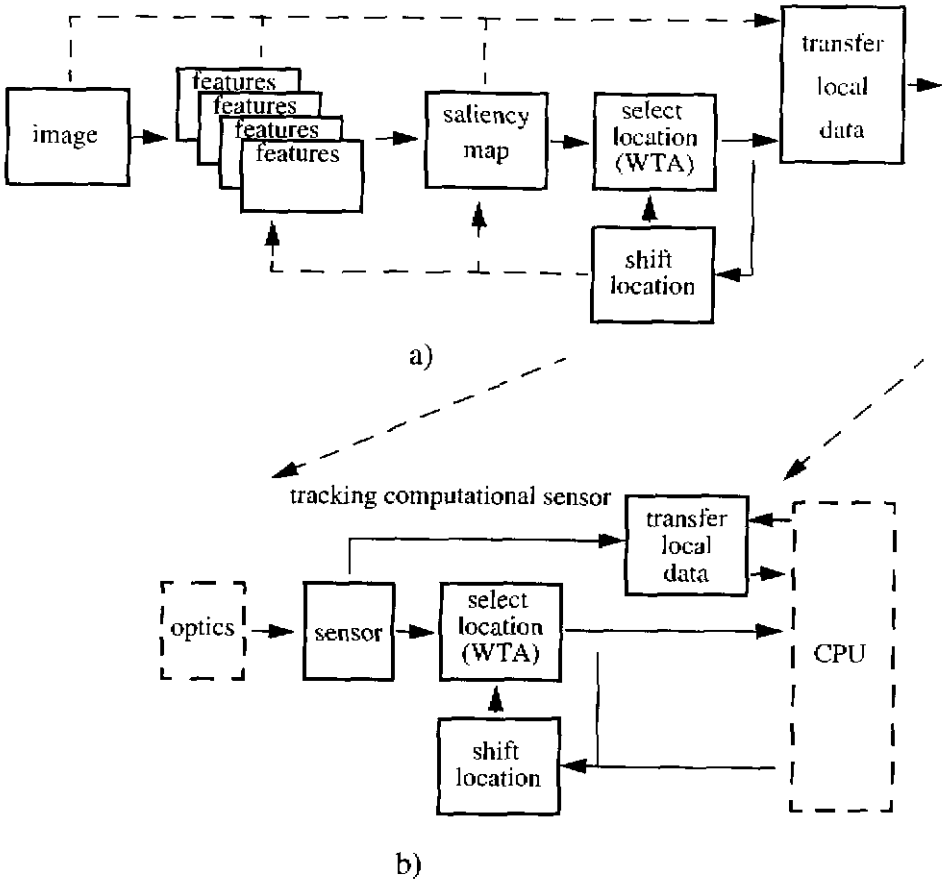


Figure 1: a) Traditional model for implementing visual attention, and b) computational sensor implementation of sensory attention.

In the tracking computational sensor we used a very compact VLSI realization of the WTA circuit originally proposed by Lazzaro [46] and Andreou [5]. The attention shifts are implemented by operating the tracking computational sensor in two modes: select mode and tracking mode. In the select mode the sensor detects the global intensity peak within a programmable active region, a subregion of the retina. (This peak is called *a feature* in the context of the tracking sensor.)[1] The sensor continuously reports the

[1] In neurophysiology the pattern of activity which activates a visual neuron is called a *trigger feature*, a somewhat controversial notion. The area of visual field in which this pattern elicits the neural responses is called the *receptive field* of the neuron. Thus, in the context of the tracking sensor the sensor itself is a neuron whose trigger feature is the peak intensity within its receptive field (i.e. the active region.)

position and intensity of the feature. By being able to program an arbitrary active region we ensure that the attention is directed towards parts of the image that are important for the task at hand. In the tracking mode the sensor dynamically defines its own active region, thus causing the sensor to ignore all retinal inputs except the currently tracked feature and its immediate neighborhood. This way our implementation ensures two things: (1) the location of attention is maintained in environmental coordinates rather then in the image coordinates, a feature important for maintaining coherent behavior in the presence of ocular or object motion, (2) the sensor eliminates interference from parts of retinal image that are irrelevant for a particular task at hand.

The WTA circuit reports the intensity of its winning input on a globally accessible wire. Therefore, by programming an active region consisting only of one cell (i.e. 1 by 1 active region), that cell becomes the winner and its intensity is reported. By scanning the trivial 1 by 1 active region around the attended location, the local data are transferred to CPU for higher processing.

The proposed implementation for the tracking computational sensory has several interesting features. The self–defined active region in the tracking mode is an example of top–down sensor/processor feedback presently missing in artificial vision systems. The global data — position and the intensity of the feature — are easily and quickly routed from the chip via several output pins. These data represent global information about the retinal image. The tracking computational sensor has numerous practical applications including continuous tracking, position estimation and structured light range imaging.

Intensity–to–Time Processing Paradigm

The other mechanism investigated is the newly proposed intensity–to–time processing paradigm — an efficient solution for massively–parallel global computation over large groups of fine–grained data [12]. Inspired by the human vision, the intensity–to–time processing paradigm is based on the notion that stronger inputs elicit responses before weaker ones. Assuming that the inputs have different intensities, the responses are ordered in time and a (global) processor makes decisions based only on a few inputs at a time. The more time allowed, the more responses are received, thus the global processor incrementally builds a global decision first based on several, and eventually on all the inputs. The key is that some preliminary decisions about the retinal image can be made as soon as the first responses

are received. Thus, this paradigm has important place in low–latency vision processing.

The intensity–to–time processing paradigm was used to implement a *sorting computational sensor* — an analog VLSI sensor which is able to sort all pixels of an input image by their intensity, while the image is being sensed. In this realization the global processor essentially "counts" inputs (i.e. pixels) as they respond. The first input to respond receives the highest index, the next input one index lower, and so on. By the time all the inputs responded, the sensor has built an *image of indices*. The image of indices represents the histogram equalized version of the retinal image. The two well know properties of such images are (1) the available dynamic range (of the readout circuitry) is equally and most optimally utilized, and (2) the image contrast is maximally enhanced. In many computer vision applications the histogram equalization is the first image preprocessing operation performed on camera images, primarily for signal normalization and contrast enhancement.

During the process of "counting" the global processor generates a waveform which is essentially the cumulative histogram of the retinal image. This waveform is one important global property of the retinal image which is reported with low latency on one of the output pins before image is ever read out.

1.5 Thesis Outline

The thesis is presented in two parts. Part I reviews foundations pertaining to the vision computational sensors, while Part II presents the proposed model.

In Part I, Chapter 2 discusses issues of vision processing and gives a brief comparative review of vision processing in brains and machines. Chapter 3 covers representative computational sensor solutions and discusses issues regarding computational sensor implementation.

Part II presents the proposed model. Chapter 4 states the goals, gives the motivation behind the proposed model and introduces two solutions: sensory attention and intensity–to–time processing paradigm. Chapter 5 introduces the sensory attention and gives the rationale behind the proposed implementation. Chapter 6 explains in detail one embodiment of the sensory attention — a tracking computational sensor. Chapter 7 introduces the intensity–to–time processing paradigm and discusses its biological plausibility. In this thesis the intensity–to–time processing paradigm is used for

sorting. The realization of the sorting computational sensor and its applications are discussed in Chapter 8. Contributions of this thesis are summarized in Chapter 9 by reflecting back to the difficulty of the problem of carrying out global operations on large groups of data, and by highlighting the efficiency of the proposed solutions for carrying out these operations within a high resolution VLSI computational sensors.

Part I:

Seeing Chips: Foundations

Chapter 2

Vision in Brains and Machines

Seemingly human vision occurs instantly and without readily discernible thought: we open our eyes and the world is perceived in all its detail. Brains are vastly complex natural systems. Even though by comparison the most advanced of today's machine vision systems are mere toys, there are enough similarities that tempt us into hypothesizing that ideas useful for understanding brains might also help us understand computers, and vice versa. A parallel computer, for example, is an information processing system made of components which mutually contribute to each other's decisions by exchanging messages over a network of connections. So is the brain. We gain insight into both systems by understanding differences between the brain's biology and today's technology as much as by dwelling on similarities.

The last 30 years of machine vision research has produced advancement along many fronts. Cameras have advanced; their resolution and sensitivity have increased, and new sensors such as uncooled infrared cameras are now

commercially available. Many recognition algorithms have been developed: from 3D model matching to artificial neural networks. Several successful practical machine vision systems have also been built: automated inspection, autonomous navigation, and medical imaging. Yet, performance of machine vision still significantly lags that of a biological system. Closing the considerable gap in processing power between machines and brains is crucial for the advancement of machine vision.

2.1 Processing in Vision

The comparative study of vision in biological and machine systems requires a rethinking of the concepts of computation. Since the mid 1940's computer science has been dominated by the von Neumann concept of serial computation, in which programs are executed one instruction at a time in a fetch–and–execute cycle. By contrast, machine vision researchers have always been aware of the highly parallel nature of the computations underlying low–level processes. Similarly, neuroscientists found evidence that in the brain billions of components are active at the same time, passing messages to one another to modify their processing. Only recently, with the advancement of VLSI technology and the increasing practice of distributed computation over networks, is the technology of computers beginning to catch up with the concepts of parallel processing in both biological and machine vision systems.

The basic structure of the vertebrate retina is that of layers of cells [19]. Furthest from the brain are the receptors — the rods and cones (approx. 10^8 receptors). Various intermediate cells are layered on top of the receptors — the horizontal cells, the bipolar cells, and the amacrine cells. These cells, in turn, provide the input to the ganglion cells, whose axons cross the surface of the retina to meet in the blind spot and form the optic nerve. The optic nerve (approx. 1.5×10^6 fibers) relays the information to the higher vision lobes of the brain. In all, the retinal ganglia are the forth or fifth order neurons from the photoreceptors, i.e. there are four or five layers of neurons in the retina processing received stimuli before the optic nerve leaves the eye [19].

The 2D structure of the retina is to a degree preserved within the representation in the intermediate neural layers. Such representations are said to be *retinotopic*[1] emphasizing their spatial nature. However, the transition from

[1] from the Greek word *topos*, for *place*.

retinotopic to abstract representations has been observed at some point as the information traverses higher centers in the brain. There seem to be mechanisms for object recognition in the brain that are not tied to the location of the object in the visual space. This distinction between the retinotopic representations in certain parts of the brain and the more abstract representations in others led to the dichotomy of the vision process into *low–level* and *high–level* vision.

It is commonly accepted that the high–level vision processes are those that apply specific *a priori* knowledge about the objects being recognized. Conversely, we refer to the processes that transform the information prior to the "knowledge–intensive processes" as low–level vision. For example, one task for low–level vision is segmentation, the process of extracting information about areas of the image (called *regions* or *segments*) that are visually distinct from one another and are each continuous in some feature, such as color, texture, motion or depth. Regions then may be considered as candidates for parts of the distinguishable surfaces of objects in the environment.

Recalling the distinction between low–level and high–level vision, one can imagine a low–level vision system as composed of vast retinotopic arrays in each of which local processors carry out the same operations in parallel. The processes of segmentation, stereopsis and optic flow can each be carried out by algorithms involving the repetition of the same local process all over an image, and can thus be characterized as parallel computation. The resulting low–level representation is then passed to more "specialized" processors, which are responsible for more abstract tasks of vision. These processors will be examining the distinctive shapes or other characteristics of the image as represented in the low–level data, and will have to pass messages to one another in order to settle on a coherent interpretation. Therefore, the high degree of parallelism is evident in both low and high level vision processing. The degree to which these parallel processes "cooperate" however requires further clarification.

Parallel vs. Cooperative and Local vs. Global Computation

Consider the layer of receptors in the retina or an electronic video camera. In a sense, these are parallel systems: a vast number of photodetectors are transducing optical information at the same time. However, individual sensing sites are not communicating, or cooperating, with each other; that is, each photodetector is only concerned with the radiation flux impinging on its own sensitive area, and may very well work as a single–point detector. This kind of parallelism we will call *non–cooperative parallelism*.

There is an abundance of evidence that the retina carries out signal processing. Some fibers in the optical nerve encode motion and others are sensitive to contrast or a particular spatial frequency. The pattern of activity that "triggers" the response of a neuron is called a *trigger feature*. The area of the visual field in which the trigger feature elicits the neural response is called *the receptive field* of the neuron. That is, the signals from the spatial extent of the neurons' receptive fields are communicated and combined into the response of the neuron. In computer vision, one analogous example is spatial filtering: in addition to its own data, a local processor in a retinotopic array produces results based upon the data from a few neighboring processors. Therefore, to produce a result the processors must cooperate and exchange data among themselves, but within limited, or local, spatial extent. This kind of parallelism is termed *a local cooperative parallelism*. Operations carried out by such a parallelism are called *local operations*.

Sometimes neurons must make decisions based on the entire retinal image. One example is the eye sensitivity adaptation. One can imagine that the system "computes" an average illumination of the retina, or finds the range of the intensities of the retinal image and adjusts the eye sensitivity accordingly. This process is not necessarily taking place in the retina, but it illustrates the need for global exchange of data for even simple decisions. In high–level vision such a need is even more obvious. The high–level processes may use knowledge of the world to postulate what objects could have surfaces positioned, shaped, or moving in the observed ways. To do this, some researchers hypothesize, each high–level process must have inputs representing *all* features that might ever appear for an object. The process is further complicated by the fact that objects can partially occlude one another or move in various directions. Thus, a visual system working in complex environments must be able to solve the problem of perspective, noise, and occlusion. (Human vision is quite successful in such tasks.) Once a preliminary interpretation of an image has been made, the vision system then has "context" and "expectations", which may activate top–down feedback paths for guiding further sensing and processing. This fascinating capability of biological vision could be explained by postulating that there are a number of quite distinct schemes or computational processes working in parallel and cooperating to form a *global* interpretation. A parallel computation that requires all (or most) of the input representations for decision making we will call *a global cooperative parallelism*, and operations carried out by such a mechanism *global operations*.

All three degrees of "cooperation" are present in vision systems and are organized in more or less hierarchical structure. The receptor layer of the

retina senses the image. This image immediately goes through several layers of neurons carrying out local operations. A retinotopic representation of an image is then relayed to the other parts of the brain where a more global exchange of information takes place. This bottom–up hierarchical structure works well as long as there is no noise or ambiguity in the retinal image. When the ongoing process of visual interpretation runs into difficulty, the top–down pathways are there to incorporate context and call upon further low–level processing and adaptation. This is dramatically demonstrated in human behavior by the dependence of eye movements on the spatial structure of the scene being viewed.

Given the complexity of the processes and information routing in bottom–up and top–down fashion, the clear anatomical and functional distinction among low–level and high–level vision begins to fade. Most researchers now tentatively agree that an interpretation is developed in stages through many levels of increasingly more abstract representations. The above discussion suggests that: (1) biological vision systems are comprised of a vast number of anatomical and functional components working in parallel; (2) these components are organized in a more or less hierarchical structure with increasingly higher level of cooperation and more abstract data representation; (3) there are bottom–up and top–down connection pathways between the processes for settling on a global interpretation of the environment.

2.2 Wiring and Information Bottlenecks in Brains

A typical biological neuron has dendrites and axons. The dendrites form a tree–like structure for receiving stimuli from other neurons. The axon is an "output wire" carrying the neural responses. The axon usually branches into an axon terminal (see Figure 2) and reaches dendrites of the neurons in the next layer. The dendrites may or may not interact with other neurons. If they do, then the dendrites are connected to the axon terminals of other neurons[1]. These connections are called synapses. A neuron receives signals from other neurons through synaptic inputs and produces activity in the axon (see Figure 3). The activity in the axon is represented by a train of "digital" pulses. The rate at which the pulses fire indicates the level of the activity in

[1] The synaptic contacts between an axon and a dendrite is called axodendritic synapse. There are other synaptic contacts: unidirectional between two dendrites — dendrodendritic synapse, bidirectional between two dendrites — reciprocal synapse, etc.

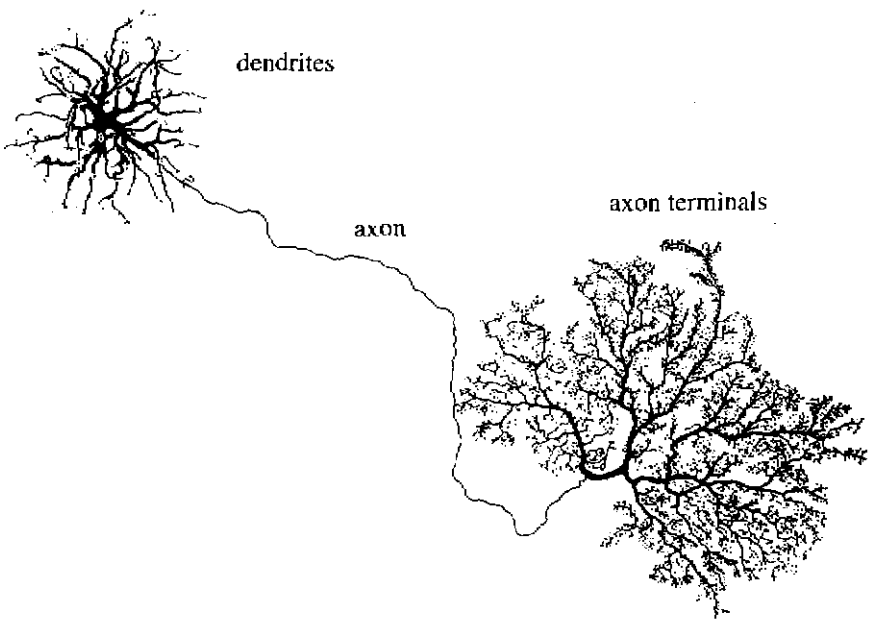


Figure 2: Light micrograph of a horizontal cells in the cat's retina [19].

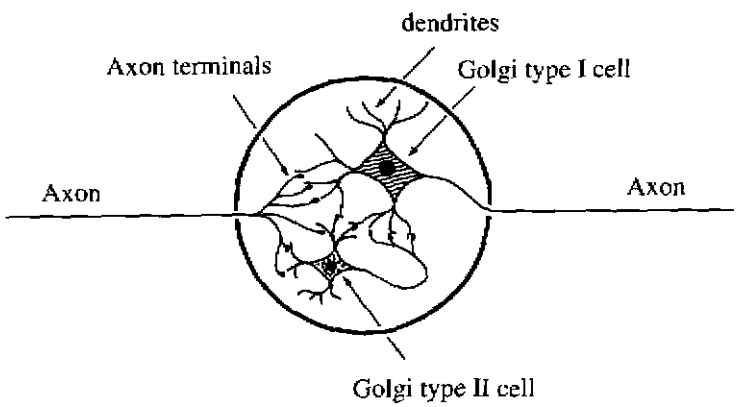


Figure 3: A highly simplified representation of a brain nucleus. The dendrites of Golgi type I cells usually only receive input from axon terminals (i.e. axodendritic synapse), but the dendrites of Golgi type II cells may both receive input and provide synaptic output (i.e.

the axon. The stimulus at the synaptic inputs that causes activity in the axon is called the *trigger feature*[1]. A neuron fires only when the input stimuli is similar to the trigger feature. Therefore, the capacity of the neuron's axon to transmit information is not fully utilized since the signals are fired occasionally; however, the energy consumption is minimized since only "useful" information is transmitted.

This discussion points to the fundamental differences in how biological and machine systems aggregate input signals. In digital systems, for example, gates with more than 4 inputs are rarely employed. (The number of inputs to a particular circuit is called *fan–in*, and the number of places to which the output goes is called *fan–out*.) The fan–in of an average neuron is 1,000 to 3,000, or even 10,000 [53]. Each input requires that a signal is routed to it. The more input signals, the more wiring is required, in both biological and artificial systems. Economizing on wire is an important priority for both nerves and chips. Yet, the biological system opted for large fan–ins. Why?

There are two plausible explanation for the large number of inputs in neurons. Suppose that a reasonably complex object in the retinal image is to be recognized. Neuron X will fire if object X is present, and neuron Y if object Y is present. The inputs to these neurons have been computed by lower neural levels in the visual path. If the objects X and Y are similar in appearance there must be a crucial feature which will differentiate the two. Therefore, the neurons X and Y must have a synaptic input representing this crucial feature. In general, the synaptic inputs to a neuron must represent all the possible features for an object, even though only a subset of them will contribute to any particular decision.

Each particular input feature can be thought of as a separate dimension in the representation of an object. A particular recognition task will have an *essential dimensionality* — a number of features that are necessary to distinguish two objects [53]. If a number of synapses is less than the essential dimensionality for an object, a unique recognition cannot be determined. This intuitive explanation is formally treated in [1]. It is shown that, under certain assumptions, the essential dimensionality is directly related to the entropy, or information, of the perceived world. This reason for a large number of inputs is called the *entropy factor* [53].

---

[1] This is a somewhat controversial notion, because a neuron will fire for a range of features that are similar to its trigger feature. Nevertheless, the neurons whose inputs closely match their trigger features will fire at a high rate, while those less well matched fire at a lower rate.

The circuit complexity theory gives another plausible explanation for the neuron's large number of inputs. Consider an N–input digital AND gate. In conventional digital technology it employs $O(N)$ transistors. The function of a single N–input AND gate can be replaced by a tree of 2–input gates. This replacement requires $O(N)$ 2–input gates. Therefore, the complexity of a digital gate does not increase when a single large input gate is replaced by a number of 2–input gates. Indeed, digital gates with fan–in larger than 4 are rarely employed, primarily due to speed considerations.

The situation is different when a large fan–in neuron must be replaced by a network of 2–input neurons. The computation of some neurons[1] can be modeled with a thresholding function. Thresholding function[2] is defined as:

$$u_i = \begin{cases} 1, \sum_{j=1}^{N} w_{ij} u_j \geq t_i \\ 0, \sum_{j=1}^{N} w_{ij} u_j < t_i \end{cases} \tag{2.1}$$

in which each input u_j takes two values, 1 and 0, and for which there exists a set of real numbers $w_{i1}, \ldots, w_{iN}$ called *weights* and t_i called *threshold*. This neuron model is known as *perceptron* and was introduced by Rosenblatt [62]. In a modified form, the perceptron is extensively used in artificial neural networks.

A 2–input perceptron can encode 16 different functions (because there are only 16 switching functions of two variables.) A network of M 2–input perceptrons thus can encode 16^M different functions. The number of 2–input perceptrons, M, needs to be at least large enough to accommodate the number of functions performed by the N–input perceptron. The N–input perceptron encodes at least $2^{N(N+1)/2+8}$ functions [56]. Thus:

$$16^M \geq 2^{N(N+1)/2+8} \tag{2.2}$$

$$M \geq O(N^2) \tag{2.3}$$

Therefore, an N–input perceptron can be replaced by a network of 2–input perceptrons but with increased complexity of $O(N^2)$. Namely, to emulate

[1] There are other neurons which take analog inputs and produce analog outputs.
[2] Threshold function is also know as majority function, linearly separable function, or linear input function.

the function of an N–input perceptron, at least N^2 two–input perceptrons must be used. This argument does not guarantee that there is a network with N^2 2–input perceptrons which can replace the N–input perceptron. It shows only that there is no smaller network that will do it. The ability of neurons to handle analog signals (through their analog weights) makes them attractive for aggregation of a *large* number of signals before making a global decision by thresholding. With a network of 2–input neurons, most of the information is lost at the intermediate discrete decisions. This loss of information must be compensated by a large number of intermediate discrete decisions. This explanation for the large number of inputs preferred by brains is called the *analog factor* [53].

Even a familiar object is never seen twice in exactly the same way: the details of the retinal images are different. Intuitively it is clear that the precision with which each input stimulus is treated is not important. The task of the brain is to make *collective* sense of sensory input. Evolution has developed the ability to make general decisions (i.e. generalize) based on a large number of inputs. If the input is incomplete, due to occlusion for example, rather then requesting higher precision with which each input is treated, the brain increases the dimensionality of the problem by introducing "context" and "expectation", or initiates eye and head movement for additional visual cues for the interpretation of the environment.

A single neuron may take many inputs, but produces a single output. If two different decisions (i.e. functions) are needed based on the same inputs, another neuron will be "wired" to those inputs, but the way synaptic inputs are combined will be different. Indeed, nearby cells in the retina may have quite distinctive trigger features (i.e. perform different functions) and yet have broadly overlapping receptive fields. This method leads to fully connected networks: each signal is wired where needed.

Even though biological neural systems have a fascinating ability to wire a large number of inputs, the biological vision is not structured to exploit the computational power of a single, fully–connected network of cells. It is rather organized into a number of areas specialized in the analysis of different aspects of the image. While these areas themselves are more or less fully connected, the connections between them present information bottlenecks. For example, there are approximately 10^8 receptors in the retina, whereas there are only approximately 1.5×10^6 axons in the optic nerve. The optic nerve, therefore, forms an early information bottleneck. Only about 2% of the information contained in the optical image focused onto the retina is transmitted through the optical nerve [20]. This dramatic data reduction

in the eye is achieved through the foveal sampling and processing by the retinal layers.

In addition to the *optical nerve bottleneck*, there is a functional bottleneck related to the visual *attention*. Figure 4 illustrates the *attention–related*

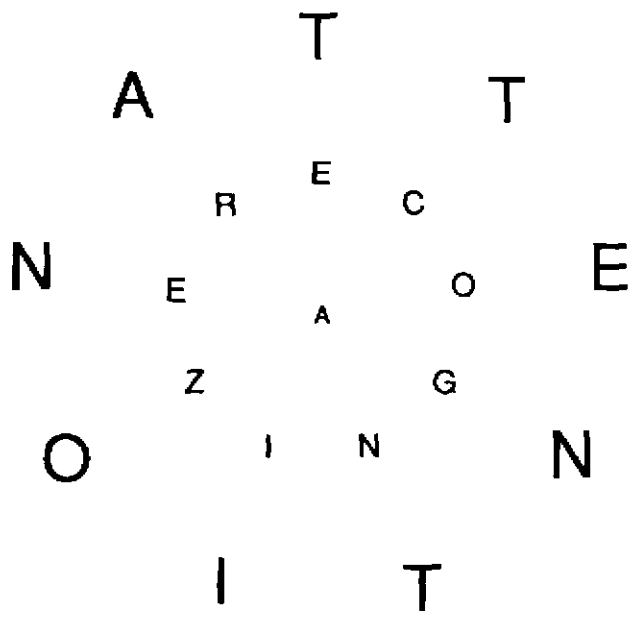


Figure 4: Illustration of the attention–related bottleneck. While fixating on the center, one's ability to recognize a characters at the periphery depends upon whether attention is directed to it. [20]

bottleneck [20]. When one fixates the letter in the center, all the letters are equally visible, as they are scaled to compensate for the decline in visual acuity with increasing distance from the center of gaze. However, in a single glance it is not possible to recognize all of the letters in the array simultaneously. In order to identify different letters in the pattern, we must direct our *attention* to the individual letters. This illustrates that there is simply too much information for the pattern recognition stage to cope with all the letters at once. Thus, by selecting a small part of the retinotopic information and dealing with it one at a time, the attention protects limited computational resources of higher vision processes from informational overload.

2.3 *Wiring and Information Bottlenecks in Machines*

Presently, neuron–like wiring is technologically not feasible in machines. Therefore, engineers have resorted to numerous "practical solutions" with varying degree of success. The most obvious choice was to trade the speed available in electronics and VLSI for the connectivity inherent in the three–dimensional organization of the nervous system. In vision sensing, for example, this trade–off resulted in *serial readout* of camera's pixel data at very *high rates* (typically 10MHz).

The serial connection between the sensor and the processor is the first (and very detrimental) *data transfer bottleneck* in machine vision systems. Images of *500 x 500 pixels* are read in *1/30*th of a second, sometimes too late when the system must cope with fast events. As the imager size increases to *1k x 1k* (or *2k x 2k*) the frame rate plunges. Some high resolution cameras provide multiple ports which could read out the four quadrants of the image simultaneously. Other more specialized imagers may provide row–parallel readout to speed up the transfer. In general, given the planar topology of today's imagers, the amount of data in an array of size N increases as N^2, while the accessible perimeter for the readout only as $4N$. Therefore, for planar structures the transfer bottleneck inevitably grows with the image size.

Once data are read out from the camera they need to be loaded and processed in a processor. The processor needs to visit data throughout the entire image before it can discard irrelevant information and focus on the important information for the problem at hand. Consequently, the processor is overwhelmed by the computational demand of vision. This causes *a computational bottleneck.*

Architectures for Machine Vision

Advances in VLSI design and fabrication technology have made it feasible to consider increasingly powerful parallel vision architectures that can be implemented in a cost–effective way. One of the potential benefits of parallel computers is the ability to scale up performance by adding more processors, hence alleviating some of the computational bottleneck in machine vision systems. There are two distinct ways in which parallelism can be applied to a problem. Parallelism can be used to solve a bigger problem in the same amount of time — called *scale–up*, or it can be used to

solve the same problem in less time — called *speed–up*. There are several measures of performance:

- *speed of processing* — operations per second,
- *latency* (or *responsiveness*) — a measure of how long it takes the system to react to an event,
- *throughput* — the rate at which data can be updated at the input and output.

In general, parallel systems can be characterized by the nature of the instruction and data streams, by the capabilities of the individual processors, and by the geometry and flexibility of the connection network between processors. Flynn proposed a classification of serial and parallel computing strategies into four general categories [36]:

- SISD — single instruction, single data stream,
- MISD — multiple instruction, single data stream,
- SIMD — single instruction, multiple data stream, and
- MIMD — multiple instruction, multiple data stream.

The SISD group includes conventional serial computers, whose inherent limitations with regard to vision tasks are well known.

The MISD category includes pipelined architectures, which received considerable attention in some commercially available vision systems such as DataCube. DataCube is a pipeline of dedicated processing modules. Each module performs one operation on a serial stream of pixel data, and feeds the results to the next module in the pipeline. Depending on the sequence of operations, the pipeline can be reconfigured and can include an image memory in between some modules. The data throughput of systems like this is designed to keep up with the data rates coming from standard video cameras: 30 frames per second. However, the *latency* through the pipeline is usually several times higher. Since only a small local portion of the pixel stream is kept in each particular module at a time (usually several scan lines), this kind of architecture is suited for local operations.

The SIMD architecture includes synchronous parallel processors in which all the processors simultaneously execute the same instructions on different data. In a SIMD machine, a centralized controller broadcasts instructions to a set of processing elements. Each processor then executes the instructions on a set of data local to the processor. SIMD architectures can take several forms, depending on the interconnection structure used. A common choice

is a fixed two–dimensional array in which each processor communicates directly with four (or eight) immediate neighbors. This type of interconnection strategy is common in image processing machines, since the structure closely matches the retinotopic structure of the image data. SIMD machines are very effective when performing regular computations which arise in some numerical and image–processing applications. Their performance falls dramatically when computation is not regular and often introduces significant transfer overhead when global communication is needed. Figure 5 shows MasPar architecture, a SIMD array with 4096 processing elements interconnected in a lateral (mesh) fashion [51]. There is a dedicated routing channel for data loading and unloading, and for global data exchange; however, the information bottlenecks are obvious.

By stacking cellular SIMD arrays and extending each processor's interconnections to include connections to the parent and sibling in the stack, one can create hierarchical (3D) architecture. If the processors from a local (lateral) neighborhood are restricted to a unique parent at the level above, then a tapering "pyramid" structure is created. Pyramid structure received considerable attention in computer vision [14], but its limitations are similar when it comes to global operations.

The MIMD machines are composed of multiple processors executing their own programs. In some machines the processors all share a common memory. In other machines each processor has its own local memory and the processors are connected together in a fixed network. The issues in these systems are (1) how to structure the interconnecting network for efficient communication between the processors, and (2) how to "parallelize" an algorithm.

Different parallel architectures have different abilities to increase performance. For example, maintaining synchronous instruction execution in a large SIMD machine is somewhat a problem: the time taken for an instruction to travel throughout the machine may be longer than the time required for execution of the instructions within the processing elements. Either the speed of operation must be restricted, or a faster (and more expensive) technology for broadcasting instructions must be used. Consequently, it is not straight forward to scale–up a SIMD machine.

MIMD machines seem to be more general, probably at the expense of being harder to program. Normally, a MIMD machine could perform the kind of regular algorithms designed for SIMD machines. The converse is not generally possible: SIMD cannot perform algorithms designed for MIMD machines.

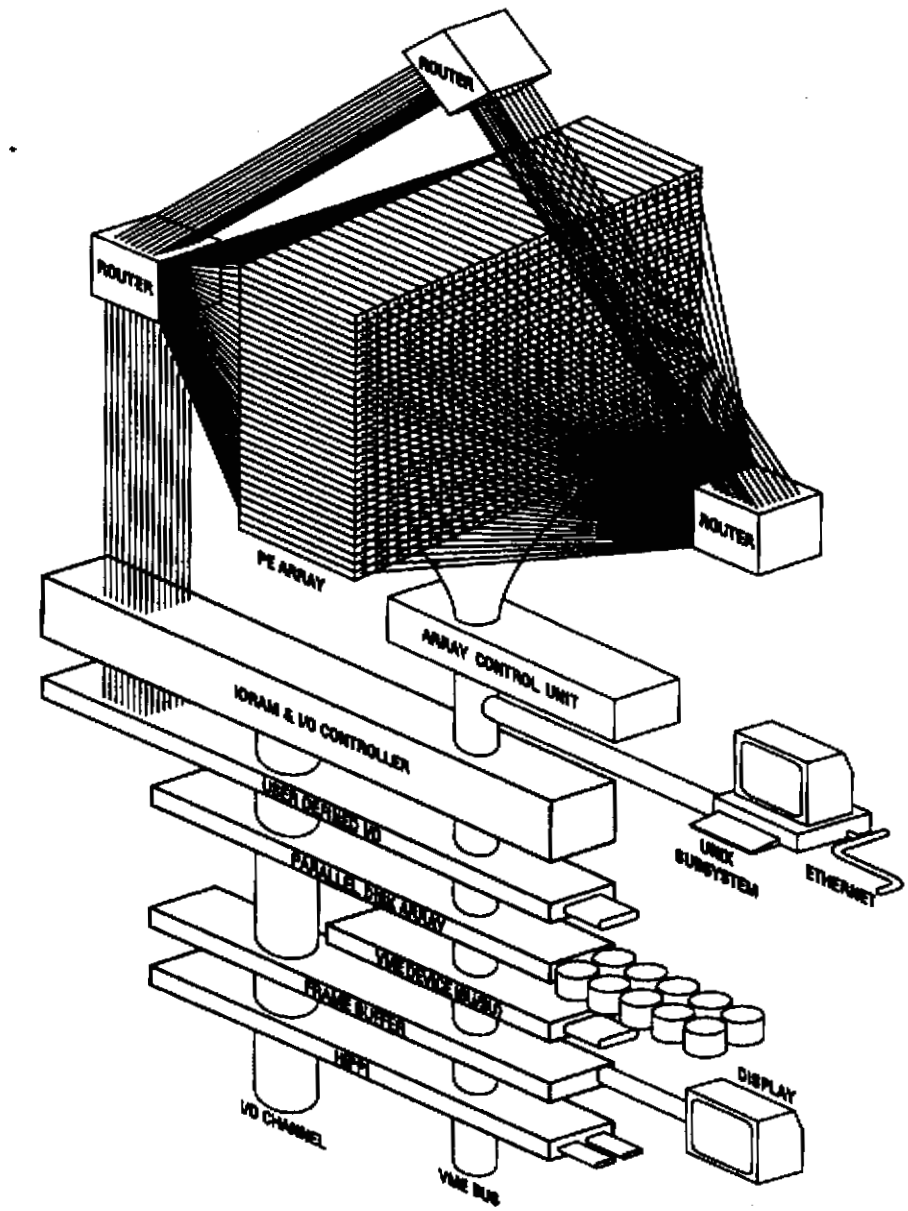


Figure 5: MasPar system architecture [51]

Another important property of MIMD architecture is the ability to scale *all* of the performance characteristics: latency, speed and throughput. Latency is the most difficult characteristic to scale. Imagine a person receiving a call on one telephone while in the midst of conversation on another. A single

person will have difficulty responding to two asynchronous calls with low latency. On the other hand, two people can easily deliver rapid response. This is analogous to the MIMD performance, which is almost impossible to emulate with a SIMD machine.

Parallel architectures outlined above have found applications in many computational intensive scientific applications. These systems are expensive, bulky and power–hungry. For vision processing, however, the single most limiting factor is the lack of efficient connectivity between the processors. This translates into information bottleneck: (1) the problem of the efficient loading (and unloading) of vast amount of image data onto the array of processors, and (2) the problem of efficient global exchange of data during computation.

Balancing Computation and Communication

For high efficiency, image data must be carefully mapped onto the array of processing elements (PE) so that the communication and computation requirements are consistent with the capabilities of the hardware. If one data item is allocated per each PE, a machine must be able to perform one operation within the same time that it communicates one data–item (e.g. receiving an operand from a neighbor). This is referred to as *fine–grain* parallelism. On the other hand, if a large number of data–items are allocated to a single PE, the communication requirements are smaller, because a PE keeps more data internally. This is referred to as *coarse–grain* parallelism.

It becomes obvious that as the grain is decreased, the communications capability becomes the limiting factor. This is why the scale–up is easier to achieve than speed–up. When parallelism is used to achieve speed–up, the computation is broken into smaller sequential segments and assigned to more processors. This in turn increases the communication requirements. For speed–up, therefore, the fine–grain parallelism is preferred, but the inter–processor communication becomes the limiting factor determining the final performance (e.g. latency and throughput) of parallel systems.

The above discussion implicitly implies a fixed 2D network in which each processing element communicates directly with a few immediate neighbors. This provides *local data communication*. However, for some vision algorithms, local communication is inadequate and it is necessary to communicate via a number of intermediate processing elements. In such cases, the communication rate becomes much too slow for fine–grain processing. In order to efficiently support a wide range of algorithms, a low–latency

machine vision system must support *global data communication* at fine–grain parallelism.

2.4 Summary

In biological vision, a vast number of components cooperate to form an interpretation of the environment. Apparently this is done in stages through many levels of increasingly more abstract representations. There are information bottlenecks between the various anatomical and functional components. The most obvious are the optic nerve bottleneck and attention–related processing bottleneck. For these bottlenecks, however, the biological vision provides two "intelligent" data compression mechanisms — retinal processing and visual attention. When the system runs into ambiguity during the process of visual interpretations, the feedback pathways may request additional processing and adaptation at the earlier stages. This goes all the way down to the sensory level. This vastly complex and effective natural system remains superior in vision tasks.

In machine vision the consistent paradigm has been that a camera "sees" the world and a computer "recognizes" the object. Implicit in this view is the separation between: a camera — a sensing (transducing) device to convert spatio–spectral–temporal phenomena to electronic signals, and a computer — a computational device to process and make sense out of data. That is, the transduced signal is read out of the sensor and digitized into the computer for processing. The separation of sensing and processing has resulted in several deficiencies in vision systems developed so far. The two most critical features missing are *low latency processing* and *sensory adaptation*.

Two main contributors to the system latency are *the transfer bottleneck* and *the computational bottleneck*. For example, when a robotic system must react to fast events it is often too late by the time the system receives an image from a standard TV camera (it takes 1/30 of a second), or by the time a computer produces an action (usually takes seconds).

Even though the data transfer bottleneck in machines may be considered analogous to the optic nerve bottleneck, and the computational bottleneck to the attention–related bottleneck, the differences are significant. Namely, the eye "intelligently" compresses data into a representation which can be transmitted through the optical nerve and, more importantly, a representation which is meaningful and well matched to the processing mechanisms

in the higher centers of the brain. There is no such relationship between the camera and the processor in today's machine vision systems. Unlike the eye, a camera transfers raw receptor data before any processing. Consequently, the transfer link might be unnecessarily burdened with communicating information which might be irrelevant to the processor. If some signal processing was available at the sensory level, early decisions could be made. These decisions could request adaptation and information refinement, so that only relevant information is transmitted to the higher processing.

All machine vision systems discussed so far are well suited for local operations, but are dramatically less effective when it comes to global operations. This is because global operations need to aggregate all input data. These data may originate at distant locations in an image and thus require wiring that is not feasible in practice. Yet, global operations are often necessary to provide those few global decisions about the environment which are necessary in guiding an intelligent machine behavior. Although nature already knows how to carry out global operations and build highly intelligent systems out of ordinary matter, she keeps it so well hidden that we may have to discover for ourselves how to do it.

Chapter 3

Computational Sensors

As VLSI technologies have matured and become more readily accessible, it has become more evident that in addition to light sensing, some degree of processing can be achieved on the same solid-state substrate. This has offered new and exciting opportunities for efficient signal processing.

Computational sensor — a sensor that computes — integrates sensing and processing on a single VLSI chip. This concept directly mimics biological systems and potentially can remove *the transfer and computational bottleneck*. Namely, on–chip routing could provide full connectivity between sensors and processors, while on–chip processors could implement massive and fine–grain parallelism which readily scales up with the image size. Finally, the clear benefit over conventional systems is the ability of computational sensors to provide feedback paths for sensory adaptation based on the result of the on–chip processing. This is something completely absent from the conventional *sense–then–process* vision systems; yet, it is so abundantly obvious in biological systems.

3.1 Computational Sensor Architectures

The computational sensor architectures that have emerged in recent years can be divided in three major groups [37]:

1. **Focal plane computational sensor:** Processing is done on a focal plane, i.e. the sensing and processing element are tightly coupled,
2. **Spatio-Geometrical computational sensor**: Computation takes place in its inherent geometrical structure and/or optical properties, and
3. **VLSI computational module:** Sensor and processing element may not be physically interleaved, but the processing module is implemented on the same sensor chip in a tightly coupled fashion.

Many existing systems would fall somewhere between the above groups. Some illustrative computational sensor solutions are presented.

3.1.1 Focal Plane Architecture

Focal plane architecture tightly couples processing and sensing hardware — each sensing site has a dedicated processing element (PE). Each PE receives the signal of its sensor. Depending on the algorithm, each PE may receive other signals including those of neighboring sensors or PE's. This concept resembles the traditional parallel fine–grain computer architectures. However, in the computational sensor realization the operands are readily distributed over an array of PE's as they are sensed by focusing an image onto the array. A physical layout of a focal plane architecture has cellular structure and is depicted in Figure 6.

Gruss, Carley and Kanade [25] [26] at Carnegie Mellon have developed a computational sensor for range detection based on light-stripe triangulation. The sensor consists of an array of cells, with each cell having both a light detector and a dedicated analog PE. The light stripe is swept continuously across the scene. The PE in each cell monitors the output of its respective photodetector, and remembers a time when the incident intensity peaks. The processing circuitry uses peak detection to identify the stripe and an analog sample-and-hold to record time-stamp data. Each time-stamp fixes the position of the stripe plane when it illuminates the line-of-sight of that cell. The geometry of the projected light stripe is known as a function of time, as is the line-of-sight geometry of all cells. Thus, the 3-D location of the imaged object points ("range pixels") can be determined through triangulation. The

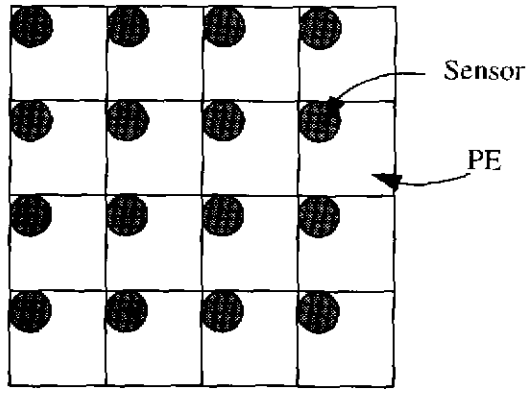


Figure 6: Layout of a computational sensor with the focal plane architecture.

spatial resolution of the range image is determined solely by the size of the cell. In the current $2\mu m$ CMOS implementation, an array of 28 x 32 cells has been fabricated on a 7.9mm x 9.2mm die. The cells in this sensor operate in a completely parallel and independent manner. It is representative of a non–cooperative parallelism.

Keast and Sodini [38] [39] at MIT have designed and fabricated a focal plane processor for image acquisition, smoothing and segmentation. The processor is based on clocked analog CCD/CMOS technology. The light signal is transduced into the accumulated charge. The neighboring PE's share their operands in order to smooth data. In one iteration, each PE sends one quarter of its charge to each of its four neighbors. The charge meets half way between the pixels and mixes in a single potential well. After mixing, the charge is split in halves and returned to the original PE, thus approximating Gaussian smoothing. If segmentation is desired, the smoothing (i.e. mixing) is prevented between those neighbors whose absolute difference is greater than a given threshold. A 40 x 40 array with cell size of about 150μ x 150μ has been fabricated [38]. This design is an example of the local cooperative parallelism.

Resistive grids are an elegant VLSI solution for spatial signal reconstruction, interpolation, smoothing and averaging [35] [41] [47]. Normally a resistive grid is distributed over the entire receptive array; thus, it receives signals from all the inputs. Depending on the exact topology of the network and the operation of the related circuitry, the resistive grids can perform global operations (case of current divider) or local operations (case of

smoothing). Computation with resistive grids is discussed in greater detail in Section 3.1.3.

Motivated by biological vision, Carver Mead at Caltech has developed a set of subthreshold CMOS circuits for implementing a variety of vision chips [53]. His pioneering design, the "silicon retina", is a device which computes the spatial and temporal derivative of an image focused onto an array of phototransistor (see Figure 7). Each cell consists of a phototransistor feeding a signal into a node of a resistive grid with uniform resistance values R. The photodetector is linked to the grid by a conductance G. An amplifier senses the voltage between the receptor output and the network potential. It turns out that the circuit computes the Laplacian of an image. Temporal derivatives are obtained by adding a capacitor to each node. Smoothing with resistive grids corresponds to the local cooperative parallelism.

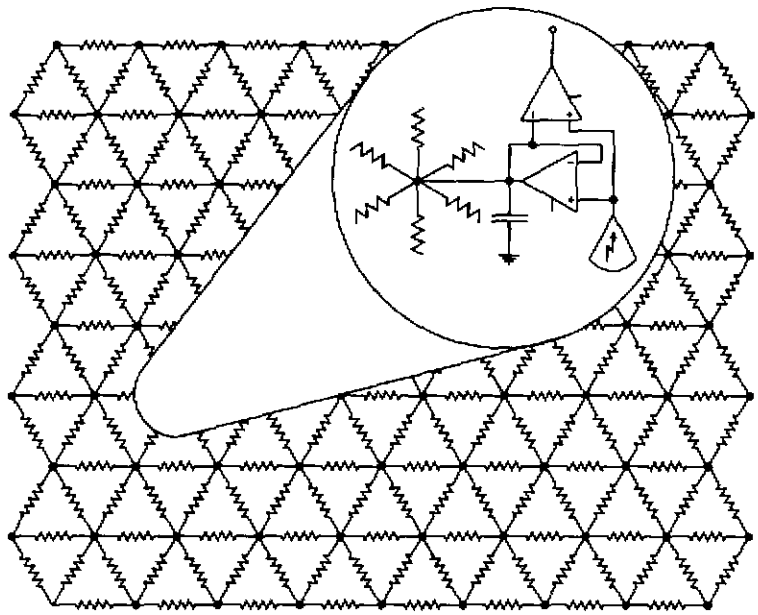


Figure 7: Carver Mead's Silicon Retina

Another example of exploiting resistive grids to achieve signal processing is an object position and orientation chip developed by Standley, Horn, and Wyatt at MIT [66]. The light detectors are placed at the nodes of a rectangular grid made of polysilicon resistors (Figure 8). The photo-current

(exceeding a global threshold) is injected into these nodes and the current flowing out of the perimeter of the grid is monitored. The injected photocurrent and the grid perimeter current are related through Green's theorem. It turns out that the perimeter current has exactly enough information to encode several moments of the object, which are in turn used for computing the object's position and orientation. An array of 29 x 29 cells has been fabricated on a 9.2mm x 7.9mm die. To work properly, the sensor requires

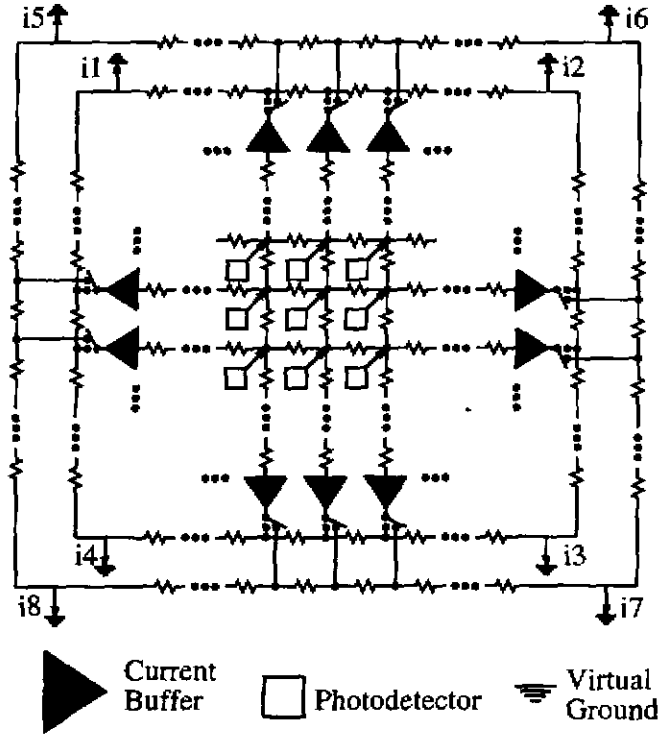


Figure 8: Horn's Position and Orientation Chip. [66]

a single bright object on a dark background. This sensor is an example of global cooperative parallelism. The resistive network (i.e. a current divider) is used to globally aggregate signals into a few averages (e.g. moments of the current distribution corresponding to the detected object).

3.1.2 Spatio-Geometric Computational Sensor

In biological vision systems it is found that sampling at different spatial resolutions is a good way to compress data. Some researchers have followed this concept by designing light sensitive arrays with new geometries, in which some degree of computing is achieved by virtue of unusual geometries, sampling grids and optical processing.

A hexagonal sampling tessellates the frequency plane more efficiently than rectangular sampling. Nature prefers the hexagonal sampling which is actually found in the mammalian retina [19]. Poussart et al. designed a 200 x 200 array with a hexagonal grid [68]. The chip facilitates parallel random access to the data in a particular local neighborhood. For rapid convolution, this local neighborhood is subsampled along three principal axes of the grid, thus reducing the data needed for convolution at the local neighborhood of each pixel. Their MAR (Multi-port Array Photo-Receptor system) performs zero-crossing detection at seven spatial frequencies in 16 milliseconds. Edge detection is computed in real time. Point-of-interest is driven by a companion microcoded controller unit. This system demonstrates advantages of direct access to the desired local image data.

A log-polar sensor developed by Kreider and Van der Spiegel [44] has a radially–varying spatial resolution: a high resolution center is surrounded with lower resolution periphery in a design resembling a human retina (see Figure 9.) This sensor must be mechanically foveated onto a region of interest. A sensor that has a high spatial resolution area, like a fovea in a human retina, is often termed a foveating sensor. By virtue of reading out the log-polar sensor and storing data into a rectangular memory, the mapping from polar to Cartesian coordinates is achieved. There is evidence in biological systems that this kind of mapping takes place from eye to brain.

Another foveating sensor has been designed by Kosonocky, et al. [73] The sensor's foveal regions can expand, contract and roam within the array of photodetectors. The chip is in essence a 512x512 square array with the ability to "merge" its pixels into regions, and output only one value for each such rectangular "super pixel". The largest super pixel is an 8x8 region. There are three modes of operation: (1) variable resolution, (2) multiple region–of–interest, and (3) the combination of the previous two modes. In variable resolution mode the entire chip has a uniform, but programmable resolution. In the multiple region–of–interest mode there are multiple active windows, possibly with different resolutions. The data are not readout from the remainder of the array. The third mode is a combination of variable reso-

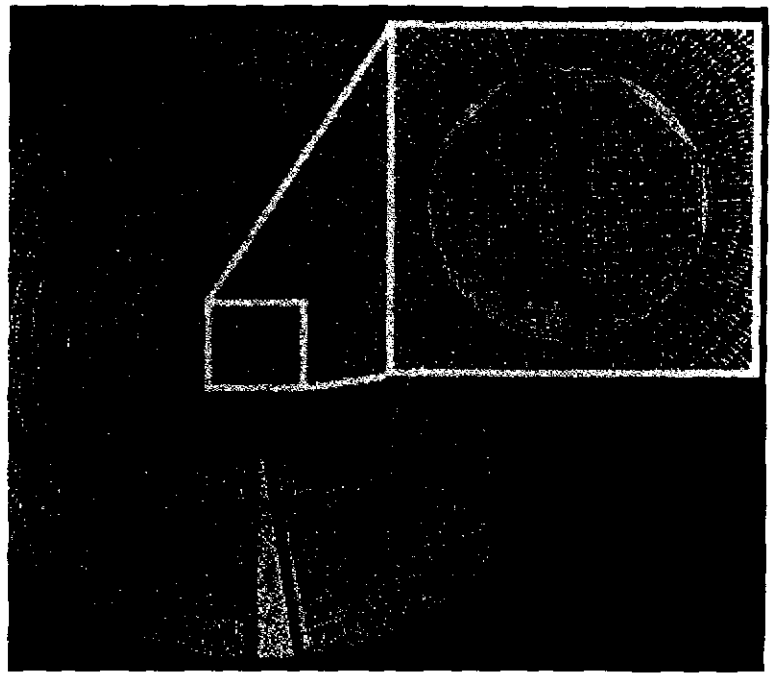


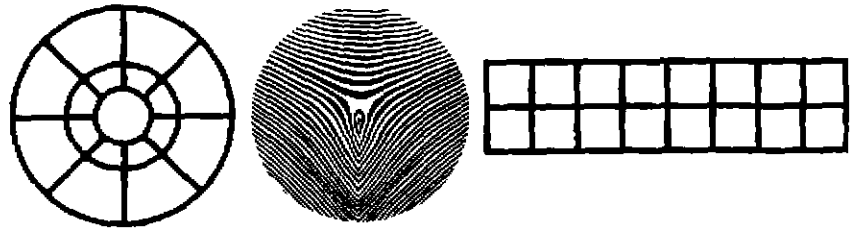
Figure 9: Log–polar foveating sensor designed at University of Pennsylvania

lution and multiple region of interest modes, and would resemble sampling of a human retina if so programmed. The design permits multiple foveae within the retina. The authors claim that there is a significant speed up in data acquisition on a variety of tasks from industrial inspection to target tracking. This sensor foveates electronically under the external control.

As VLSI microlithographic techniques have advanced, the fabrication of binary optical devices has become possible [70]. By etching desired geometrical shapes directly on the surface of an optical material, a designer can produce diffractive optical elements with properties that were previously impossible to achieve. Such devices introduce some simple optical processing before the light is detected. Veldkemp [70] developed a micro lens array in which each lens is only 200 microns in diameter. One application of such an array would be to focus light onto tiny photodetectors and therefore save the silicon area for processing hardware. Some of the first applications of the idea using refractive lenses are already in the market: Hitachi FP-C10 HI-8 video coders use a micro lens CCD, and Sony XC-75

video camera doubles the sensitivity to f8 @ 2000Lux by their HyperHAD CCD structure that uses micro lenses.

Experimentation with binary optics showed that it can generate virtually any transformation of an optical wave front. The first application that used this new capability was a binary optical component that optically mapped the log-polar plane to the cartesian plane (Figure 10). This device, in effect, samples images at log-polar geometry and then optically routes them for sensing on a Cartesian grid. This way an optical log-polar foveating sensor is achieved with a rectangular sampling grid camera. This technology has great potential for the optical routing of signals.



a) b) c)

Figure 10: Log-polar image mapping using binary optics: Optical transformation converts concentric circles (a) to straight lines (c). (b) is a contour phase map of the optical element that forms the image [70].

3.1.3 VLSI Computational Module

There is a class of computational sensors in which processing elements are not physically interleaved with sensors: however, the processing is performed on a tightly coupled module integrated on the same chip. This is the *VLSI computational module* computational sensor architecture. Figure 11 shows a possible chip topology for this architecture. The computational module is used when:

- there is not enough space to accommodate complex PE's between the sensing sites, or
- there is no obvious parallelism in the data, or even the data are not result of sensing, but the algorithm naturally maps to physical processes in silicon and is efficiently carried out only in VLSI.

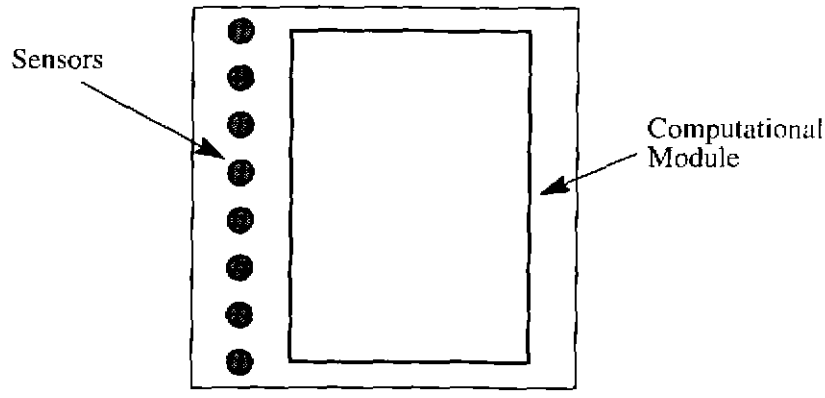


Figure 11: VLSI computational module architecture

There are two analog optimization techniques which are well suited for implementation in silicon: regularization with resistive grids based on the Maxwell's Minimum Heat Theorem, and the gradient descent method. Both of these techniques are formally presented in [35] [41]. An intuitive illustration of these techniques is given below.

Consider the problem of fitting a 2D surface through a set of noisy and sparse measurements. It is obvious that infinitely many surfaces can be fitted through the sparse data set. One way to regularize the problem is to impose a smoothness constraint by penalizing the surface derivative, and then solving the resulting quadratic variational problem. It turns out that these problems can be solved within simple linear resistive networks by virtue of the fact that the electrical power dissipated in linear networks is quadratic in current or voltage. The minimum heat theorem states that in a steady state, a circuit will dissipate minimum energy. Thus, each resistor will tend to reduce the voltage difference on its nodes. In the process, all the resistors collectively arrive at the overall surface reconstruction. The surface reconstruction network is shown in Fig. 12. Noisy measurements (u_i) are loaded onto the network, and the surface is "solved" as node voltages ($\hat{u}_i$). Intuitively, the resistance value R determines the level of surface smoothness, while the value of conductance G determines the allowed variance between noisy measurements and surface solution. At nodes where measurement is not present, G is set to zero.

There is another class of optimization problems formulated in terms of minimizing an objective function $J(\bar{x})$. The independent variable $\bar{x}_0$, for which the objective function is minimized, is the solution of the optimiza-

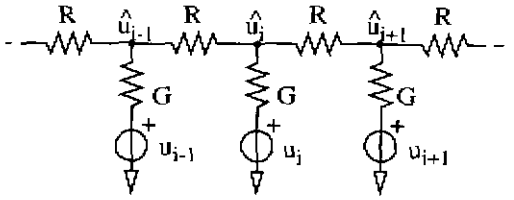


Figure 12: A resistive ladder network for signal reconstruction/interpolation.

tion. In the well known gradient descent method this solution is reached from an initial guess by iteratively taking steps in the direction of a local gradient. The "length" for each step is proportional to the magnitude of the gradient. When the solution is reached the gradient is zero and no further steps are taken. In a digital computer the iterative steps are discrete and this method may run into various problems such as slow descent, overshoot, or instability. On the other hand, if the method is implemented in an analog network, the solution is found as a steady state of a continuous dynamic system. For its implementation, one may imagine a current source feeding current to a capacitor. For nonzero currents, the capacitor's voltage will increase or decrease, depending on the current's polarity. When the current is zero, the voltage on the capacitor remains unchanged. Imagine now that the voltage on the capacitor represents a variable x. This voltage is used in a circuit which computes the (momentary) gradient of the objective function $\Delta J(x)$. Now, this gradient, in the form of a current, is fed back to the capacitor, and the voltage, or variable x, gets updated. This process, of course, proceeds in a continuous fashion, and when a steady state is reached, i.e. the gradient becomes zero, the voltage on the capacitor represents the solution x_0 of the optimization.

Numerous computational sensor chips use the resistive network smoothing or gradient descent optimization technique for computation. For example, an early and impressive effort to detect image motion has been made by Tanner [83]. He successfully built and tested an 8x8 pixel chip which produces a single uniform velocity averaged over the entire image. His chip reports *Vx* and *Vy* velocities which minimize the least square error derived from the optical flow constraint equation [34]. The solution is found using the gradient descent method.

The zero crossings of the Laplacian of the Gaussian (LOG) operator, ∇G, are often used for detecting edges. The LOG operator could be approximated by the difference of two Gaussians (DOG). Bair and Koch [7] have

built an analog VLSI chip that detects zero–crossings in the detected one–dimensional image. The image is detected by a one–dimensional array of 64 photoreceptors. The image is then smoothed by two separate resistive networks, producing two increasingly smooth copies. The chip then takes the difference of these two copies, and reports the zero-crossings in this difference. An adjustable threshold on the slope of zero-crossings can be set to cause the chip to ignore weak edges due to noise.

The detection of discontinuities in motion, intensity, color, and depth is a well studied but difficult problem in computer vision. Standard smoothness assumptions smooth out noise in the data but also tend to smooth away the discontinuities. To combat this problem, Harris and Koch [30] have invented "resistive fuses". Like a normal household fuse, a resistive fuse operates as a linear resistor for small voltage across it, and as an open circuit for large voltage drops. Thus, the network of resistive fuses smooths the noisy data, but "breaks" for large differences between neighbors and thus selectively allows discontinuities. A 20x20 rectangular grid network of fuses has been demonstrated for smoothing and segmenting test images which were scanned onto the chip [30] [31].

3.1.4 Comparison of Computational Sensor Architectures

All of the computational sensor architecture described above have advantages and limitations. The focal plane architecture resembles traditional fine–grain parallel computer architectures: each data item is assigned to a different processing element (PE). The focal plane architecture is well suited for local operations because it is relatively easy to connect immediate neighbors. However, the silicon area take by the processing element grows as the complexity of the processing element increases. For planar structures, therefore, the focal plane architecture cannot simultaneously accommodate complex PEs and maintain high spatial resolution.

When there is not enough space to accommodate a complex PE within a cell/pixel the VLSI computational module can be used. There are several computational sensor designs that use dense linear array of photodetectors with all the processing electronics integrated to the side of the array[7] [33]. In most of the cases, due to the size of a PE this was necessary rather then desirable solution.

Sometimes, the computation mechanisms native to processes in silicon are implemented in a VLSI computational module. Examples of such computational mechanisms include charge transfer processing, large resistive grids signal reconstruction, and large gradient descent optimization. The input

data may be entered by electrically scanning them into the sensor, rather then by sensing them. Electronic loading of massive amount of data presents a new set of problems ranging from slow speed to signal noise and degradation.

In spatio–geometrical computational sensors the computation takes place in its inherent geometrical structure and/or optical properties. It is advantageous to use focusing optics for some simple processing such as blurring. In addition, the size, shape and placement of photodetectors results in some information processing. For example, a large photodetector *averages* illumination over its active area.

Which architecture is used depends on a particular application. Most often, a particular architecture is a hybrid of the three described architectures. For example, a focal plane architecture can have a resistive grid distributed over the entire array of sensors. This resembles both the focal plane architecture and the VLSI computational module: the parts of the resistive grid are integrated within each cell/pixel but perform a meaningful function only in conjunction with parts from other cells/pixels. As another example, if the array of photodetectors is not uniformly distributed over the chip area, a particular computational sensor has features of the spatio–geometric architecture.

When implementing global operations, ideally the global processor is interleaved with the array of sensors. This way each sensory signal has direct access to the processor allowing large data throughput between the sensor and processors. The function implemented by such a global processor must allow simple and compact VLSI implementation. For example, a single wire running and connecting all the cells/pixels can function as a global adder if the operands submitted by each cell is a current. In another example, if a global processor can broadcast information globally to all the cells if that information is presented as a voltage on a single global wire. Some of these concepts are used in our computational sensors presented in Part II of this thesis.

3.2 Computational Sensor Issues

Issues regarding computational sensors include:

- VLSI technology
- choice of an algorithm,
- applications

3.2.1 VLSI technology

The technologies available in VLSI are CMOS, Bipolar and BiCMOS. CMOS is characterized by very dense packaging, low power consumption and high input impedance. Good switching properties make CMOS well suited for both digital and hybrid circuits. Driven by the silicon market, it is also widely accessible and relatively inexpensive technology. CCD is implemented in MOS technology. Bipolar technology is characterized by low noise and fast circuitry, but consumes more power and takes more silicon real estate. It is not as accessible to the wider research community as it probably should be. BiCMOS combines the advantages of both CMOS and Bipolar technologies. Semiconductor materials other than silicon are also available. GaAs compound yields very high speed circuitry and is well suited for electro-optical applications. GaAs technology is less available, however, and is more expensive.

The trend in VLSI is toward smaller device geometries. This produces both smaller and faster digital circuits, hence more functionality per unit area. The feature scaling, however, is not as beneficial to the analog circuitry as to the digital. Most of the active devices are designed at a given size and scaling would not preserve the desired functional features. Analog MOS circuits benefit more from improvements in fabrication process quality. Factors such as oxide quality and thickness, or tighter control of threshold voltages would greatly benefit analog circuit performance. However, dominated by digital markets, these parameters are not necessarily optimized for analog performance.

Great interest has been shown for possible optical signal communication between stacked chips which would lead to 3D VLSI. This could be accomplished with the availability of silicon-compatible semiconductor IR emitters and detectors [69]. This technique would also require integrated optics, such as binary optics. Alternatively, a conducting vias could be developed for making distributed point-to-point electrical through–substrate

connections [58]. Micro fiber–optics, such as coherent bundles, could be used to route data in parallel from module to module. As already mentioned, the diffractive optical components and holographic technique have great potential for optical signal routing. The optical approach has the advantage of possible optical processing in the data transmission itself, but has the disadvantage of high power consumption and heat dissipation. This technology is not yet been developed enough to become accessible to the wider research community.

Analog vs. Digital

Both digital and analog circuits can be implemented in VLSI technology. The analog approach can be further divided into continuous-time (i.e. unclocked) and discrete-time (i.e. clocked) processing. Which one to use depends on the particular chip application, but several general remarks are in order. Compared to digital, the traditional disadvantage of analog electronics is its susceptibility to noise, which yields low precision. The source of this noise can be on-chip switching electronics, which requires special considerations for hybrid designs. Also, analog electronics does not provide efficient long-term storage; typical longest storage times are about one second. On the other hand, digital processing requires A/D and D/A conversion, which usually imposes limitations on the total circuit speed. In summary, analog electronics is characterized by:

- high speed,
- low latency,
- low precision (typical 6 to 8 bits),
- short data storage time (typical 1 second),
- sensitivity to on-chip digital switching; and
- long design and testing process.

In general, analog hardware takes less chip area than the digital mechanism of the same functionality, and so far seems to be the preferred choice for a Computational Sensor.

Analog VLSI offers two interesting advantages for Computational Sensor computation. First, the physical properties of the solid-state layers and devices can be exploited to yield elegant problem solutions, such as resistive grids and sheets. The second interesting advantage of the analog VLSI is charge-domain processing which is best exemplified by CCD technology. It

offers an area-efficient mechanism for transferring the data. Creative processing schemes can be invented to process the data in charge-domain as data are transferred. CCD technology has already provided many signal processing examples, and recently it has shown several useful examples of *integrated* sensing and processing [24] [27] [38].

3.2.2 Algorithms

Since there are practical limitations in available circuits and architectures, algorithms must be carefully selected or invented to match the underlying hardware for maximum performance. The circuitry may have limited precision and long-term storage for a given algorithm. Similarly, an inappropriate algorithm may require a processing element that is too large to fit. Therefore, suitable algorithms need to be robust to noise, and must exploit a significant level of parallelism without requiring significant storage capacity, space, or inter-processor data transfer.

The computing mechanisms, such as those offered by the resistive grids, clearly requires that new algorithms be adapted or invented. In fact, historically speaking, ideas for quite a few vision algorithms came from the considerations of the laws of physics, and yet a particular implementation is the outcome of *reducing* those laws to a form suitable for a serial digital computer implementation. Some of the original thinking based on laws of physics can be inherently and naturally embedded in the semiconductor material, offering new exciting implementations.

Optical methods are a good way to compute in parallel. For example, a simple blur is a way of low-pass filtering an image. Holographic techniques and advances in binary optics apparently seem to offer an interesting opportunity.

3.2.3 Applications

The VLSI computational sensor offers exciting possibilities. One must be careful, however, when deciding which applications will benefit from such an implementation. At the present state of technology, analog VLSI design is a lengthy process. Until technology allows much denser circuits (or 3D structures), it is clear that there is not enough room to fabricate a complex processing element at each photo site. In such a case, processing and sensing must take place on separate, but tightly (preferably on-chip) coupled modules. The cost of transferring data must be minimized in order to justify the use of VLSI over conventional systems. CCD row-parallel transfer is

one way to keep the transfer at a reasonable speed. Some algorithms may not directly exhibit parallelism in the focal plane and often require significant local data storage at each processing element. In stereo algorithms, for example, optical signals are to be combined from two different focal planes. In this case, data are read out and processed on a separate computational module. It is valid in this case to question whether it is justifiable to invest the effort in developing a custom VLSI processor, or if it is better to use a digital computer. If data must be transferred from the sensor to the processor through low-bandwidth serial channel, then there is no obvious reduction in data flow. Whether the limited precision of analog VLSI is offset by its low power consumption and small size depends on the particular application.

Part II:

Proposed Model

Chapter 4

An Overview

The task of visual perception, in one view, is to infer relevant properties of objects in the environment that have caused a particular spatio–temporal–spectral pattern of light detected by an array of photoreceptors. This view implies that vision must provide a veridical representation of the external world. However, the viewpoint pursued in this thesis is that the job of vision is not to provide a machine (or animal) with a representation of the world *in abstracto* but to provide the machine (or animal) with the *information it needs to interact* with the world around it. In this view it is essential that the vision is able to reliably and quickly react to the events in the environment.

4.1 Main Goal

The main goal of this thesis is to design vision computational sensors which reduce the latency in a vision system, and provide top–down adaptation feedback for more reliable performance. This goal is further driven towards producing a task–oriented self–contained machine vision component that

can be used by the machine for reliable and timely interaction with the environment. In order to attain this goal, the problem of implementing global operations in computational sensors is addressed.

In the context of this thesis the global operations are important for two reasons. First, in perception it seems that each important decision is a kind of global, or overall, conclusion about a perceived world. These conclusions are often what a machine needs for coping with a task at hand. The global operations thus can be considered to produce the ultimate goals of the vision processing for the coherent interaction with the environment. Second, global operations produce *a few* quantities for the description of the environment. Therefore, these quantities can be quickly transferred and/or processed to produce an appropriate action for a machine. In addition, the results of the global operations can be used within the computational sensor in top–down sensory adaptation thus directing a further processing for more reliable performance. This concept is illustrated in Figure 13 showing block diagrams of conventional and a computational sensor approach.

The most interesting computational sensor solutions so far have been inspired by biological vision — notably the retina. For example, Carver Mead's group at Caltech has developed a number of *retinomorphic* computational sensors which directly mimic computational processes in the retina. Their design principles have stressed the importance of similarities with the biological retina: continuous logarithmic detection, local gain control, local adaptability, and encoding the local spatial and temporal derivatives [10]. These are local operations and the retina itself carries them out. Local operations use operands within a small spatial/temporal neighborhood of data and lend themselves to graceful solutions in VLSI. While computationally demanding, the local operations produce large quantities of preprocessed image data. In animals this information is relayed to the brain via the optical nerve; then, the global signal aggregation and *decision making* takes place in the higher centers in the brain. The vast number of fibers in the optical nerve is a luxurious transfer bus which machines cannot replicate at the moment. Therefore, communicating the retinotopic representation from an artificial retina to the decision making processor presents the transfer bottleneck. This fact renders most of the computational sensors less effective in robust low–latency machine vision systems which must make fast decisions about the environment.

Global operations, on the other hand, result in few entities for description of the environment. If computed at the point of sensing, these entities could be routed from a computational sensor through a few output pins without causing the transfer bottleneck. This information will often be sufficient for

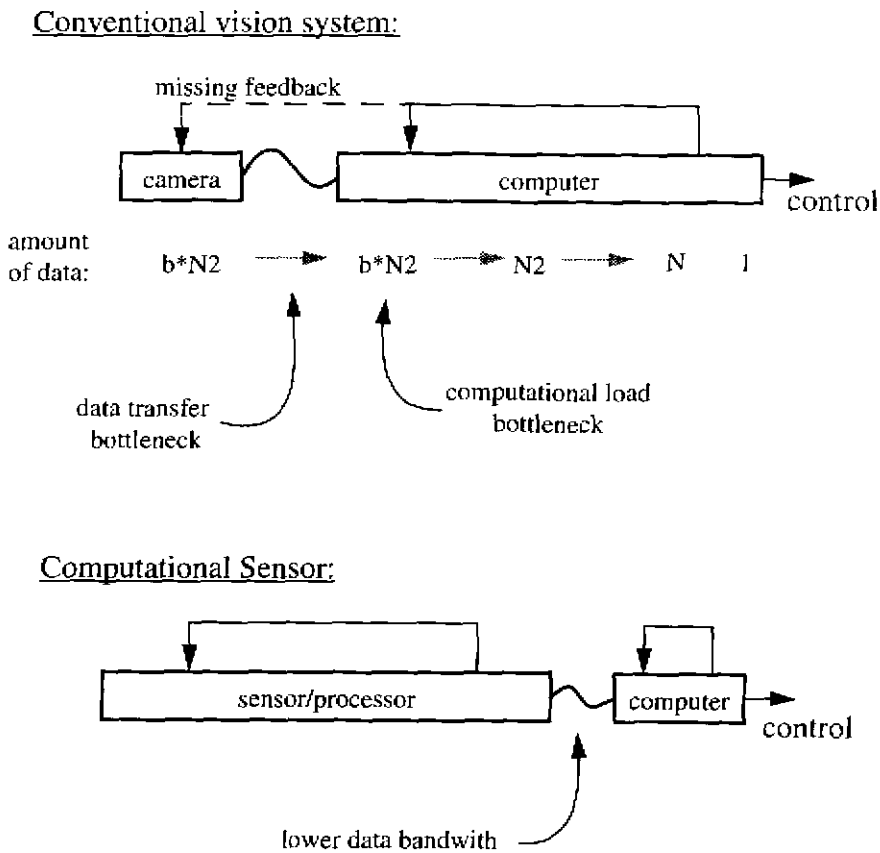


Figure 13: Block diagram of a conventional vision system and a computational sensor.

rapid decision making and the actual image does not need to be read out. The computed global quantities can be used to update (local and global) internal information and adapt further sensing and processing. This top–down link is completely absent in conventional vision systems, and would hardly be achieved with purely retinomorphic chips performing mainly local computations.

Carrying out global operations within a computational sensor at the present day technology clearly requires that some of the higher level processing is integrated within the computational sensors. This may require distinctly non–biological implementations and, therefore, the departure from the one–to–one similarity with the *anatomy* of biological vision (e.g. the retina, optical nerve, brain). Nevertheless, exploiting differences as much as similarities provides valuable insight into the problem. Evolution certainly likes

pipes, valves and pumps, but has little use for wheels — an invention that has revolutionized human civilization.

4.2 Implementing Global Operations

Implementing global operations in parallel systems has been the subject of extensive research in both computer engineering and computer science. The main difficulty with implementing global operations comes from the necessity to bring together, or aggregate, all or most of the data in the input data set. This global exchange of data among a large number of processors/sites quickly saturates communication connections and adversely affects computing efficiency in parallel systems — parallel digital computers and computational sensors alike. For example, in order to sort a set of input numbers, each number must be communicated to the sorting processor. If this is done sequentially, it takes a long time before the result is available. On the other hand, sending all of the input numbers simultaneously requires: (1) a wire for each number, which would enormously proliferate the amount of wire needed, and (2) a sorting processor which can handle all the numbers at once, which is also not practical.

Let us recall for a moment several features of signal aggregation and processing in biological systems:

Large number of inputs. Neurons take a large number of inputs, each representing an identifying feature. Humans seem to find that the more distinguishing features an object has, the easier it is to recognize it. The opposite is true for a computer: the more features an object has, the harder it becomes for the computer to quickly recognize it.

Essential dimensionality. Recognition cannot be achieved if the number of inputs is less than the essential dimensionality for the object. For two similar objects additional inputs representing one or more *crucial features* are necessary to distinguish the objects and determine the recognition.

Precision vs. cooperative parallelism. Precision by which each input is handled in neurons is not essential; if a decision cannot be made the dimensionality of the problem is increased. Therefore, the lack of precision is compensated by massive fine–grain cooperative parallelism.

Trigger feature. The neuron's trigger feature on its input produces activity in the axon. This activity is represented by a train of pulses. The rate of firing the pulse shows how well the input feature matches the neuron's trigger feature. Normally, several patterns of stimulation similar to the trigger feature produce the response, but the trigger feature produces the highest rate.

Visual Attention. By selecting a small part of the retinotopic information and dealing with it one at a time, the attention helps compress data at higher levels of vision processing.

Signal transmission. The pulses in the neural axons occur infrequently and are used only to transmit "interesting" information. This scheme conserves energy, but leaves the transfer capacity of the axon underutilized.

Nature prefers a large number of signals. Noisy or incomplete data as well as the imprecise processing of neurons is compensated by massively parallel cooperative computation over a large number of inputs. The object recognition is robust, because a decision making neuron (or network) seems to have a large number of inputs representing all features that might ever occur, even though *only a subset of them may appear at a time* and contribute to any particular decision. When there is a danger of overloading the processing capacity, the attention seems to be the mechanism which "funnels" features *one at a time* to the decision making neuron.

Inspired by these observations the two different mechanisms for implementing global operations in computational sensors are proposed: (1) *sensory attention*, and (2) *intensity–to–time processing paradigm*.

Sensory Attention

The sensory attention is based on the premise that salient features within the retinal image represent important global features of the entire data set. Then by selecting a small region of interest around the salient feature for subsequent processing, some global conclusions about the retinal image can be made. The sensory attention eliminates extraneous information and allows the processor to handle small amount of data at a time. This prevents computational and transfer bottlenecks. The sensory attention clearly mimics the process of visual attention in higher centers of the brain: a small interesting portion of the retinal image is selected to which the higher level processes can be restricted. Unlike eye movement (i.e. *overt* shifts), the attention shifts (i.e. *covert* shifts) do not require any motor action, but occur

internally on a fixed retinal image. For this reason, attention shifts are faster and play an important role in low–latency vision systems.

It is interesting to note that foveating computational sensors try to emulate this kind of data compression. For example, Van der Spiegel's log–polar sensor [64] samples images within fovea with high acuity, while maintaining sparse representation of the surrounding. This sensor requires motor action for foveating and simulates the overt shifts. Kosonocky's foveating sensor [73] allow programmable fovea within the retinal image; therefore, it eliminates the need for mechanical action and simulates covert shifts. Another related solution is a random access to the image data. For example, Laval's MAR sensor [68] attends to and reads only a small local portion of the retinal image, the part that is necessary for the local convolution performed in the global off–chip processor. In either case, these computational sensors act as special cameras and the mechanism which guides the location of the attention is missing.

A saliency map is used to encode conspicuousness throughout the image. A winner–take–all (WTA) mechanism has been proposed for guiding the attention to the most salient parts of the retinal image [42]. The WTA circuit determines the identity and magnitude of the strongest point [23] in the saliency map. The WTA mechanism has been also proposed for other brain functions: several cells would respond to the pattern of activity, but the winner would determine the perception.

We used an extremely compact VLSI realization of the WTA circuit [46] [5] and built a computational sensor implementing the sensory attention. The saliency map is delivered to the sensor optically by focusing it onto the array of photodetectors which feed the WTA circuit. The features which attract attention are peaks in the saliency map. When natural images are focused onto the sensor— a trivial saliency map — the features that attract attention are bright spots in the retinal image. Therefore, this particular embodiment of the sensory attention is called the *tracking computational sensor* — a VLSI sensor that attends onto and tracks bright spots in the retinal image.

We implemented the attention shifts by operating the tracking computational sensor in two modes: select mode and tracking mode. In the select mode the sensor detects the global intensity peak within a programmable active region, a subregion of the retina. (This peak is called *a feature* in the context of the tracking sensor.) The sensor continuously reports the position and intensity of the feature. In the tracking mode the sensor dynamically defines its own active region, thus causing the sensor to ignore all retinal

inputs except the currently tracked feature and its immediate neighborhood. This ensures that interference form the irrelevant information within the retinal image does not interfere with the currently attended information, i.e. the information important for the task at hand. In the tracking mode the sensor effectively remains locked on the selected feature and maintains the location of attention in the environmental coordinates.

The self–defined active region in the tracking mode represents an example of sensor/processor feedback presently missing in artificial vision systems. The global data — the position and intensity of the feature — are easily and quickly routed from the chip via several output pins. Inherent in this implementation is the ability of the sensor to provide random access to the image data if needed. The image data can be read from a random location within the retinal image including the vicinity of the feature being tracked. The features position and intensity, as well as image data selected by attention represent global information about the retinal image. Further rationale regarding the implementation of the sensory attention is presented in Chapter 5. Details of the tracking computational sensors are discussed in Chapter 6.

Intensity–to–Time Processing

The other mechanism for implementing global operations is the newly proposed *intensity–to–time processing paradigm* — an efficient solution for massively parallel global computation over large groups of fine–grained data [12]. Inspired by the human vision, the intensity–to–time processing paradigm is based on the notion that stronger signals elicit responses before weaker ones. Assuming that the inputs have different intensities, the responses are ordered in time and a global processor makes decisions based only on a few inputs at a time. The more time allowed, the more responses are received, thus the global processor incrementally builds a global decision based on several, and eventually on all of the inputs. The key is that some preliminary decisions about the retinal image can be made as soon as the first responses are received. Therefore, this paradigm has an important place in low–latency vision processing.

The intensity–to–time processing paradigm has been used to implement a *sorting computational sensor* — an analog VLSI sensor which is able to sort all pixels of an input image by their intensity while the image is being sensed. In this realization the global processor essentially "counts" inputs (i.e. pixels). The first input to respond receives the highest count/index, the next input one count/index lower, and so on. By the time all the inputs

responded, the sensor has built an *image of indices*. The image of indices represents the histogram equalized version of the retinal image. The two well known properties of such images are (1) the available dynamic range (of the readout circuitry) is equally (i.e. most optimally) utilized, and (2) the image contrast is maximally enhanced. In many computer vision applications the histogram equalization is the first image preprocessing operation performed on camera images, primarily for signal normalization and contrast enhancement.

During the process of "counting" the global processor generates a waveform which is essentially the cumulative histogram of the retinal image. This waveform is one important global property of the retinal image which is reported with low latency on one of the output pins before image is ever read out. The details of the intensity–to–time processing paradigm are presented in Chapter 7, while the details and applications of the sorting computational sensor are discussed in Chapter 8.

Chapter 5

Sensory Attention

The main argument that has been used to explain the need for selective attention in brains is that there exists some kind of processing limitation in the visual system: "the attention protects the limited processing resources from the information overload" [3]. Empirical evidence shows that when attention is attracted to a location, one can notice improvement in oculo–motor performance such as the ability to more rapidly identify, locate and respond to certain stimuli.

Considering the limited communication and processing resources in machines, the attention selection certainly seems attractive: a small portion of the retinal image is selected and brought to a global processor for higher level processing. Indeed, the importance of selecting the relevant information from an image is now widely acknowledged in machine vision and some form of attention mechanisms (e.g. selecting a correctly sized window within the image) are often employed in practical applications.

5.1 Traditional Models of Attention

A number of psychophysical studies suggested a two–stage model for object localization and recognition in human perception [43]. The first stage is described as preattentive in which low–level processes compute in parallel over the entire retinal image. The second state — the attentive process — selects a small portion of the preattentive information for subsequent high–level processing. The selection process proceeds as a *spotlight* sequentially moving across the visual field. The spotlight metaphor was first proposed by Neisser [57] who suggested that the process of sequential attentive "analysis" should be termed "synthesis", since this process builds more complete representation of the environment as the attentive process recognizes more and more features. This is the most accepted model of visual attention in cognitive science. To apply attention selection in machines, several issues must be solved: (1) the problem of selecting an "interesting" location, (2) the problem of shifting to another location, and (3) problem of transferring local data for further processing.

5.1.1 Koch and Ullman's Model

In a very influential paper [43], Koch and Ullman address these issues. The selection process utilizes a *saliency map* that encodes conspicuousness or the level of interest throughout the retinal image. The saliency map can be derived from image features including the intensity, color, spatial and temporal derivatives, motion, and orientation. For selecting a location of the attention within the saliency map a *winner–take–all* (WTA) mechanism has been suggested. The WTA ensures that only one location is selected among many potentially interesting locations. The WTA is not responsible for information processing, it only determines which area of the retinal image should be relayed to the global processor for further inspection.

The problem of shifting to another location is somewhat more challenging. It is observed in humans that interesting visual stimulation initially (i.e. during the first 100ms) captures the attention but later (i.e. after 300ms) has inhibitory effects which can last up to 1.5 seconds [54]. The inhibitory effect prevents the subject from returning to previously visited locations. The inhibition is "stored" in environmental coordinates rather than in image coordinates; therefore, the reliable operation is maintained even in the presence of ocular or object movement. The attention shifts can be initiated on a voluntary basis, caused by telling the observer the location of a target, or they can be automatic caused by the onset of a visual stimulus.

For shifting to another location Koch and Ullman's model allows the saliency of the currently attended location to decay, even if the visual stimuli creating the saliency remain present. This will release the WTA mechanism and allow it to converge to another location. Either a *local* or *central* inhibition mechanism for initiating decay is possible. The local mechanism causes the saliency to decay some time after the WTA has converged to a particular location. In the central mechanism, once the attended portion of the retinal image is relayed to the central processor, a signal is sent back which inhibits the conspicuousness of the currently attended location. The local inhibition mechanism mimics the automatic attention shift, while the central mechanism can initiate voluntary attention shifts.

Once the WTA is released to converge to another location, two rules are suggested: (1) shift to another conspicuous location within the close spatial proximity of the most recently attended location (i.e. proximity preference), and (2) shift to another location whose saliency is similar to the saliency of the most recently attended location (i.e. similarity preference).

5.1.2 Attention–for–Action Model

Koch and Ullman's model of attention is to a certain degree geared towards object recognition tasks. For example, the proximity preference together with the local inhibition mechanism automatically moves the "spotlight" of attention across the conspicuous features of an object (e.g. discontinuities, contours, etc.) and funnels information to the global processor which incrementally builds a representation for the complete recognition of the object.

Allpot suggested, however, that attention goes beyond protecting the limited processing resources during complex object recognition: attention is needed to ensure behavioral coherence (i.e. *attention–for–action*) [3]. Since visual perception is the means that allows a subject to interact with the environment (e.g. manipulate, avoid, etc.), it must produce actions consistent with the subject's goals. Selective processing is necessary in order to isolate the information that defines parameters for the appropriate action. For example, to catch a moving object (among many other objects) the information specific only to that object determines the action. Information about other objects in the visual field must be kept from interfering with the goal of catching the target object, even though other objects may influence how the target object is caught. In other words, attention allows the target goal to be completed by masking the interference from the irrelevant information,

but allows the action to be modified or diverted if new important events occur.

5.2 *Proposed Implementation*

The attention–for–action model is in close agreement with our goal of producing reliable low–latency computational sensors which make global decisions for coherent interaction with the environment. It is not hard to imagine that if the attention is allowed to arbitrarily roam from one location to another, as suggested by Koch and Ullman's model, it may take a long time before the global processor comes across the *relevant* information for an appropriate action. The situation becomes worse in the presence of noise which could misguide the attention and divert it from the information needed for the ultimate goal. We need more control over attention shifts, possibly employing the central inhibition mechanism in combination with the voluntary focus of attention directed toward desired goals. For robust operation such shifts must maintain the location of attention in the presence of ocular or object motion.

5.2.1 *Location Selection*

A very compact VLSI realization for the WTA circuit [46] [5] is used for selecting the location of the attention. The circuit receives the two–dimensional saliency map, identifies the strongest feature in it, and outputs the feature's location and magnitude. Figure 14 illustrates these steps for one–dimensional case.

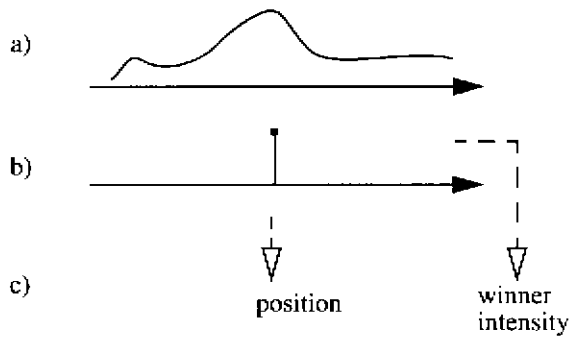


Figure 14: Illustration of the WTA circuit operations in one dimension: (a) sensing a saliency map, (b) finding a winner and generating a binary representation, and (c) computing the position of the non–zero output.

5.2.2 Location Shifts

The two dimensional WTA circuit locates the absolute maximum in the entire saliency map. Without significant increase in circuit complexity, it is possible to inhibit portions of the saliency map and restrict the activity of the WTA circuit within a programmable active region — a subset of the circuit's receptive field. The active region is programmed by appropriate row and column addressing. This corresponds to the central inhibition control suggested by Koch and Ullman.

There are two modes of operations: (1) select mode, and (2) track mode. In the *select mode*, the active region is defined by the external addressing (Figure 15a). The active region can be of arbitrary size and location. In *tracking mode*, however, the sensor itself defines a small (e.g. 3 x 3) active region centered around the most recent location of the attention (Figure 15b).

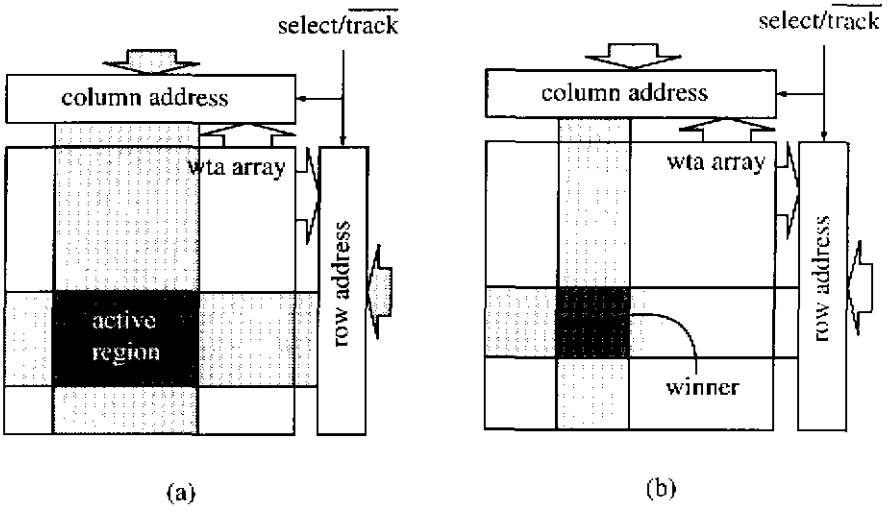


Figure 15: Modes of operation for the sensory attention computational sensor: (a) select mode, and (b) tracking mode.

In practical applications, there are often several strong features in the saliency map. All of these features could be good candidates for attracting the attention. The *select* mode directs the attention towards a feature that is useful for the task at hand. For example, a user may want to specify an initial active region, aiding the sensor to attend to the relevant local peak in the saliency map. Then, the *tracking* mode is enabled for locking onto the selected feature. The tracking mode ensures that the location of the attention

is maintained in the environmental coordinates even in the presence of motion.

5.2.3 Transferring Local Data

Once the relevant conspicuous point has been localized in the saliency map, the local data from the attended vicinity must be transferred to the global processor for decision making. The local data originate from any early representation including: image data, early features used for building the saliency map, or the saliency map itself.

The circuit for sensory attention described so far has access to the saliency map only. With the suggested implementation, it turns out that the local information from the saliency map can be easily transferred to the global processor. In fact, the magnitude of the localized feature in the saliency map is continuously reported to the global processor, as it is inherently measured by the WTA circuit [46] [5]. If the surrounding points are also needed, the global processor can program a trivial 1 x 1 active region at the desired location. It inhibits all other input of the saliency map, and forces the WTA circuit to choose that particular point as the winner and report its magnitude to the global processor.

5.3 Summary

The proposed implementation for the sensory attention exhibits several interesting features. It performs a global operation over the saliency map and produces few global results: the position and magnitude of the selected saliency feature. These global results can be routed off–chip with low latency via few output pins. Furthermore, in the tracking mode the global results are used internally for programming 3 x 3 active region and therefore providing a top–down feedback for securing robust performance in tracking the attended feature.

We have not discussed how the saliency map could be computed. Perhaps when the VLSI technology allows 3D structures, a stack of retinomorphic computational sensors for computing local features for use in the saliency map can precede the attention chip. Still the location of the attention can be used internally down this stack of chips for guiding sensing and processing in upcoming time interval. This represents a fast top–down feedback which is not currently present in the conventional vision systems.

At the present state of technology we built a chip that receives the saliency map optically: the saliency map is focused onto an array of photodetectors which feed the WTA circuit. The problem of creating an appropriate saliency map now becomes the problem of creating an appropriate optical pattern. A bright spot in such a pattern is considered salient and will attract attention. When in the tracking mode, the chip will lock–and–track the bright spot as the spot moves in the field of view. This chip is called a *tracking computational sensor*. The details of its implementation and its applications are presented in the next chapter.

Chapter 6

Tracking Computational Sensor

We built a computational sensor which implements the sensory attention as introduced in the previous Chapter. In this implementation the two–dimensional saliency map is delivered optically by focusing an image onto the chip. The easiest application of this sensory attention chip is obviously with natural images, in which case the salient features are bright peaks in the image. This particular embodiment of the sensory attention is called a *tracking computational sensor.*

The tracking computational sensor is a smart optical position detector: it automatically locks and tracks the bright spot in the image while ignoring the background. Applications for this sensor range from depth sensing to human–computer interaction systems.

6.1 Feature Selection

The tracking computational sensor detects an image, finds a feature therein and continuously reports the location and intensity of that feature. The feature is a local maximum (i.e. peak) in the image. The mechanism underlying the feature detection is the winner–take–all (WTA). The tracking computational sensor has a cellular focal plane architecture: each cell is comprised of a sensor and a small processing element. The cells are connected to a common global wire. The voltage on this wire arbitrates among all the cells and promotes the maximum input to win.

6.1.1 Winner–Take–All Circuit

Our design is based on a WTA circuit originally proposed by Lazzaro et al. [46]. The circuit has few components and its realizations is very compact. It has been used as a component in several VLSI sensory systems including systems that perform visual stereopsis [48], and visual motion detection [33].

The original WTA circuit is shown is Figure 16. Currents $I_1 \ldots I_N$ are the inputs, while currents $J_1 \ldots J_N$ are the outputs of the WTA circuit. The cell receiving the largest current $I_k = max(I_1 \ldots I_N)$ responds with non–zero output current $J_k \neq 0$ while other cells respond with zero currents, i.e. $J_i = 0$, for $i \neq k$.

The circuit operates as follows. The source coupled transistors $T_{11} \ldots T_{1N}$ share a common current source I_c. A two–transistor version of this circuit is a well–known differential pair. A transistor with a greater gate voltage than

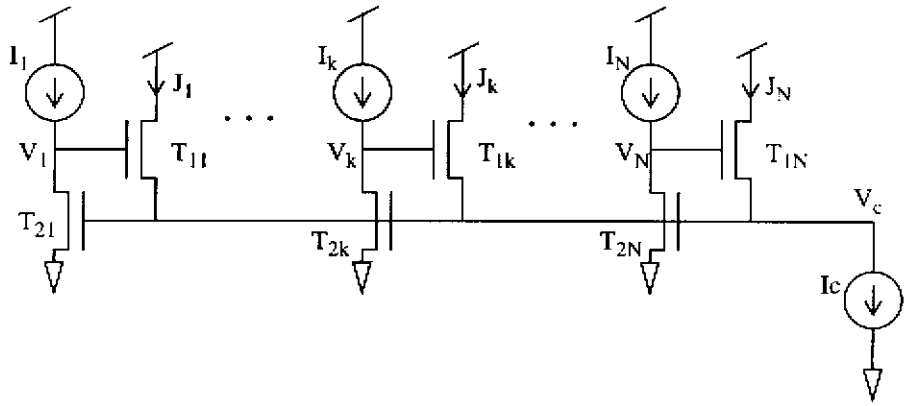


Figure 16: Schematic diagram of the winner–take–all circuit. Shaded area indicates one cell.

others sinks a larger portion of the current I_c. In essence, the source coupled transistors performs the winner–take–all (WTA) operation on the gate voltages. In order to sink all of the I_c, and be the winner, a particular gate voltage has to be several tenths of millivolts higher then the next highest input voltages. This is a significant difference which limits the resolving power of the WTA. The resolving power of a WTA circuit is defined as the minimum required difference between the winner and the next highest input for which the winning transistor sinks all of the I_c. A high gain inverter circuit comprised of an input current source I_k and a transistor T_{2k} provides additional gain; therefore, small changes in the input currents produce large variations at the gates of the source coupled transistors. This increases the resolving capability of the circuit and promotes a clear winner for small input current differences.

The inverter transistors are all connected to a common wire carrying a voltage V_c. This is the key feature that enables self–biasing of the inverters and prevents the saturation. The voltage on the common wire, V_c, is set up by automatic arbitration on the input currents. To illustrate the winning/losing behavior of the circuit, consider a network of two cells shown in Figure 23. If the input currents are equal, due to the symmetry of the circuit, the current I_c would split equally among T_{11} and T_{12}.

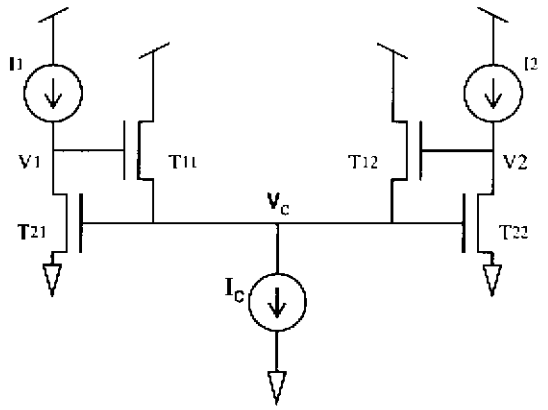


Figure 17: Two–cell winner–take–all circuit.

Suppose the input current I_1 is a bit larger then I_2, i.e. $I_1 = I_2 + \Delta I$. Since both transistors T_{21} and T_{22} have the same gate voltage V_c, the difference in input currents (i.e. drain currents of T_{21} and T_{22}) must be compensated by changes in the drain voltages V_1 and V_2. This can be seen in Figure 18 showing an I_d/V_{ds} curve for a particular common V_c. Since the input currents (i.e. drain currents) are different, the operating points for the tran-

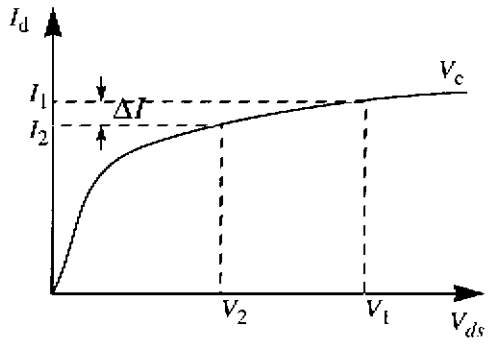


Figure 18: I_d-V_{ds} curve of transistor T_{21} and T_{22}.

sistors T_{21} and T_{22} lie on different locations on the curve. In order to compensate for smaller I2, the drain voltage V2 must be smaller in respect to V_1. These voltages are gate voltages for the T_{11}/T_{12} differential pair. The higher voltage V_1 will cause the transistor T_{11} to sink more current then T_{21}. For a given ΔI, the higher the transistor resistance (i.e. the flatter the curve) the greater the difference between V_1 and V_2. When the circuit settles, V_2 is pushed to the linear (i.e. steep) region of the curve, while V_1 holds transistor T_{21} in saturation. Therefore, the peak current establishes and holds the common voltage V_c. For small input currents, like those produced by light detection, the transistor operates in the subthreshold region. In that case the voltage V_c is the logarithm of the winning input current:

$$V_c = V_o \log\left(\frac{I_1}{I_o}\right) \tag{1}$$

where I_o is the process parameter and $V_o = kT/q\kappa$. Therefore, the intensity of the winner can be accessed globally by monitoring the voltage on the common wire.

6.1.2 Output Quantities and Extensions

The output of the WTA arbitration used by Lazzaro are voltages $V_1 \ldots V_N$. Namely, if $I_k = max(I_1 \ldots I_N)$, then V_k is a logarithmic function of I_k; if $I_j \ll I_k$, then $V_j \approx 0$. Indeed, assuming subthreshold regime the winning voltage in the two–cell example is [46]:

$$V_1 = V_c + V_o \log \frac{Ic}{I_o}$$
$$V_2 \approx 0 \tag{2}$$

The second quantity on RHS of Equation 2 is V_{gs} of T_{11}. However, T_{12} turns off when its V_{gs} has reached zero. This occurs when $V_2 \cong V_c$. Since common wire voltage V_c is non-zero, T_{12} turns off for $V_2 > 0$; therefore, current through it reduces to zero earlier then voltage V_2. If the drain currents of T_{11} and T_{12} are outputs, rather than the voltages, then the winner is obtained for smaller ΔI. Therefore, by monitoring drain currents of the source coupled transistors, the winner is obtained for smaller ΔI. This is equivalent to saying that the voltage difference between V1 and V2 is amplified by the transconductance of the differential pair T11/T12. This observation was also made by Andreou *et al.* [5]

The DC performance of the two–cell WTA circuit of Figure 23 has been simulated. The findings are graphed in Figure 19. The top graph shows volt-

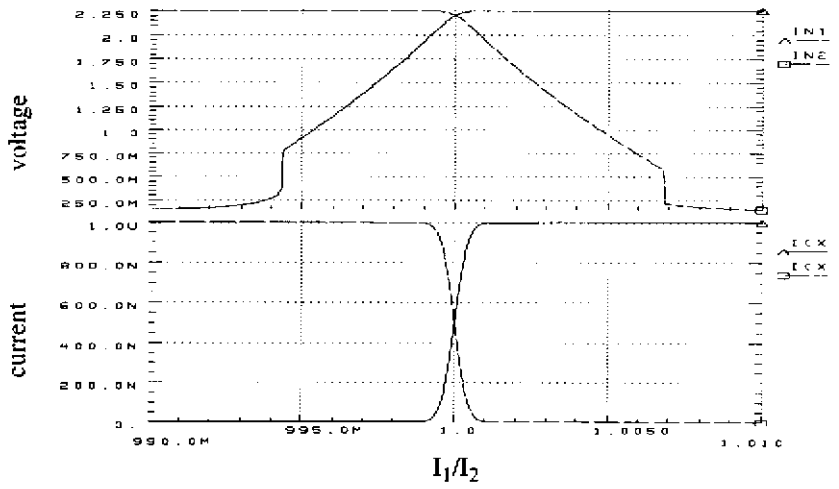


Figure 19: Winning/losing behavior with voltage output and current output.

ages V_1 and V_2 as the ratio of input currents is varied. The voltage of the losing cell has reduced to zero when the input currents differ for about 0.7%. However, shown in the bottom graph it is seen that a clear winner among output currents J_1 and J_2 is obtained for input current difference smaller then 0.1%. Currents J_k are therefore better indicator of the winning cell.

Based on this critique of the original circuit, we propose the following output quantities to be used with the winner–take–all circuit:

- the logarithm of the winning input is observed on the common bus as voltage V_c,
- the winning cell k signals with non–zero current, $J_k = I_c$, through the T_{1k}.

6.1.3 Input Quantities and Photo detection

The input current sources can be realized as photodetectors (see Figure 20).

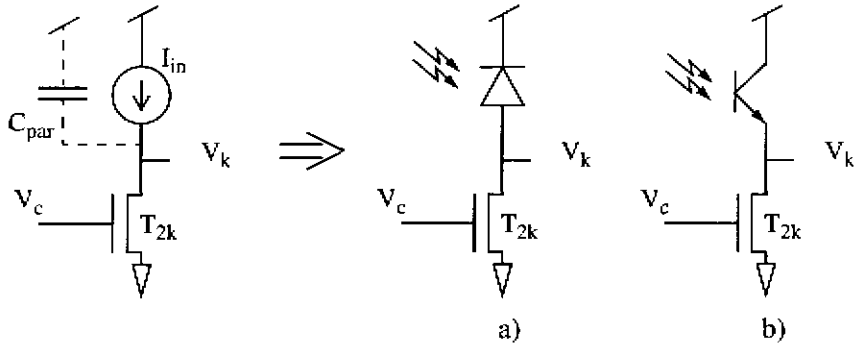


Figure 20: Input current source realized as (a) photodiode, and (b) photo transistor.

Photodiode or photo transistor are available in a VLSI process provided by Mosis. Photodetectors have relatively large parasitic capacitance; therefore, for low light levels the response is too slow. One remedy is to buffer the photocurrent. The current buffer maintains (nearly) constant voltage on the parasitic capacitance, while conveying the photocurrent to the processing circuitry. This works well for photodiodes.

Photo transistor provides somewhat higher currents due to the beta amplification of the photocurrent. Since the photocurrents are small, the beta is nonlinear and varies from 1 to 30 for the ranges of average room illumination[1]. Photo transistors tend to be slower than photodiodes, because the buffering strategy does not restrict voltage variations on the floating base. Consequently, during fast transitions, a portion of the photocurrent is wasted on charging and discharging the base capacitance. Whether the

[1] The beta was measured on a vertical NPN transistor built in the 2μ ORBIT CMOS process provided through MOSIS.

current gain obtained by the photo transistor prevails over its slower response depends on a particular application.

6.1.4 Localization of the Winning Input

Only the winning cell responds with non–zero current. This is effectively 1–of–N binary encoding. A digital on-chip decoder easily converts this code to any other binary code such as a natural binary or BCD code. In addition, there are efficient analog means for winner localization [17].

One example is shown in Figure 21. The outputs from each WTA cell are

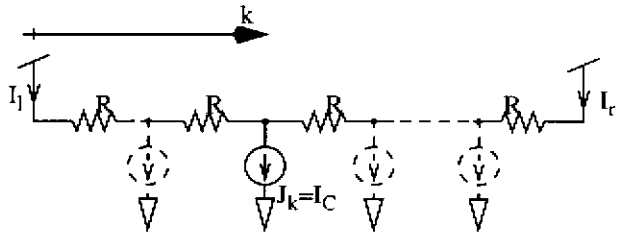


Figure 21: Resistive network for position detection.

connected to nodes of a linear resistive network. The WTA ensures that only one of these currents is non–zero (i.e. I_c). The goal is to determine the *location* of this current. Suppose I_c is sunk at the node k. The network behaves as a current divider, and the current I_c is split into I_l and I_r given as:[1]

$$I_l = \frac{N-k}{N} I_c \qquad I_r = \frac{k}{N} I_c \tag{3}$$

If currents I_l and I_r are measured, position k is found as:

$$k = \frac{I_r}{I_c} N \qquad I_c = I_l + I_r \tag{4}$$

If currents are injected at several nodes, we can show, using superposition, that Equation 4 yields the centroid of the injected currents. This is an efficient method to compute and access a useful global quantity [17] [35] [72].

6.1.5 Two–Dimensional Array

The WTA cells can be physically laid out in a two–dimensional array. Still one of the cells wins and provides non–zero output current. Using the

[1] Both ends of the network are held at the same potential.

method of projections [34], the position of this current in two dimensions is found by solving two one–dimensional problems. A projection is the sum of the object intensities along a line (i.e. line integral) perpendicular to the projection axis. The x and y centroid coordinates of the winner are found as the position of non–zero current in the horizontal and vertical projections respectively.

This concept is implemented as shown in Figure 22. Two copies of the output current are summed into the horizontal and vertical bus respectively.

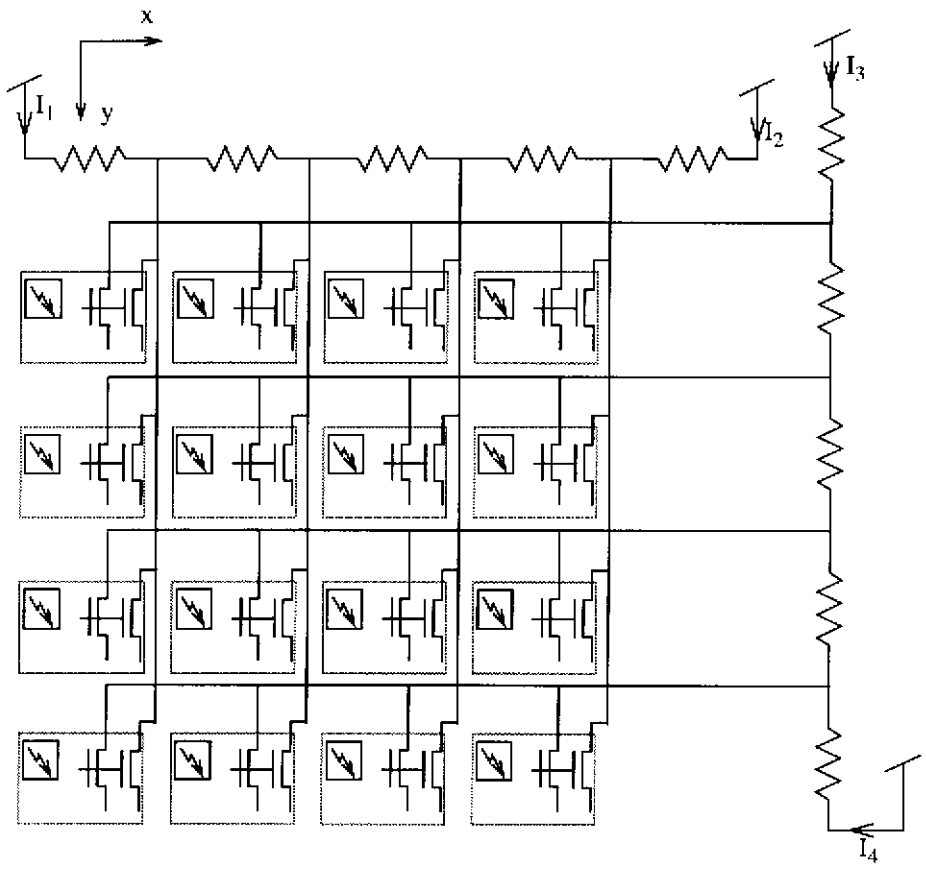


Figure 22: Two dimensional WTA computational sensor.

The total current in these buses represents the desired projections onto the x and y axes. Then, two linear resistive networks are used at the periphery of the array to locate the winner in x and y direction.

6.1.6 Dynamic Behavior

The temporal response of the WTA circuit is important when tracking moving features. Figure 23 shows the two–cell WTA circuit exposing para-

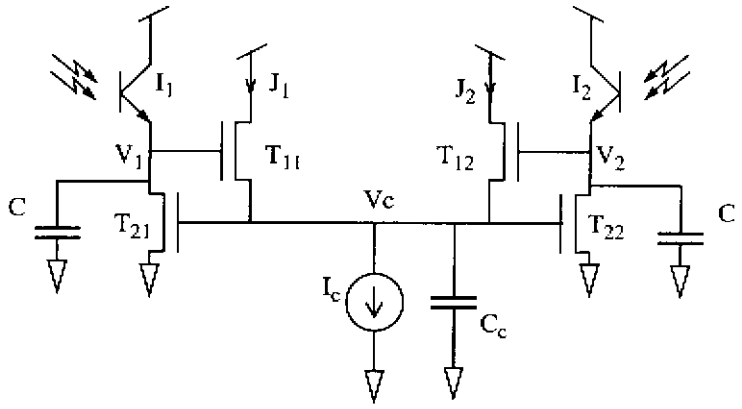


Figure 23: Two–cell WTA circuit showing parasitic capacitances.

sitic capacitances. Capacitor C models the cumulative parasitic capacitance of the gate of T_{1k}, the drain of T_{2k} and the capacitance of the photodetector. The capacitor C_c models the parasitic capacitance of gates of T_{1k} and the common wire. The transistor T_{1k} is a long channel device for improving the drain resistance, which in turn improves resolving capabilities of the WTA circuit discussed earlier. Thus, the gate area and its capacitance is moderate. The common wire reaches every cell throughout the chip and therefore its capacitance is large.

A first–order analysis of a two–cell WTA circuit has been done in [46]. It has been found that the circuit is stable if:

$$I_c > 4I_p\,(C_c/C), \quad \text{for} \quad I_1 \approx I_2 \approx I_p \tag{5}$$

Even though the ratio C_c/C is large and increases with number of cells, the detected photocurrents I_p are small, and the stable condition is easily maintained by controlling a common current I_c. In experimentation with a two–dimensional prototype with about 1000 cells I_c = 1μA sufficed. For larger arrays, the current I_c needs to be scaled up proportionally. For example a 200 x 200 array would probably require $I_c \approx 40\mu A$ which is easily attained within the chip.

Using the first–order small signal analysis, Lazzaro *et al.* [46] predicted dynamics of the voltages V_1 and V_2. The time constant for the winning cell

is CV_o/I_p where $V_o = kT/q\kappa \approx 40mV$ is the process parameter. For the losing cell the time constant is CV_e/I_p, where $V_e \approx 50V$ is the Early voltage and is the measure of transistor's resistance. The WTA circuit exhibits a large signal excursion and has a highly nonlinear gain. Nevertheless, the small signal analysis used in [46] confirms the intuition: for a cell to win or lose the parasitic capacitance C must charge and discharge by the photocurrent I_p. For average room illumination the photocurrents are very small, less then $1nA$. The WTA circuit is therefore slow.

Insight into the large signal dynamic behavior is obtained through simulation under following conditions. The photodetectors capacitance in each cell is $0.5pF$. The capacitance of the common wire simulates 10,000 WTA cells ($C_c \cong 50pF$). The common current is $I_c = 10\mu A$. At $t = 100ms$ the photocurrent to the first cell makes a step transition from 75pA to 25 pA, while the photocurrent to the other cell transitions from 25pA to 75pA. This effectively simulates a step transition of an optical feature from one cell to the next. We observe how quickly the first cell loses and the second one wins.

The dynamic behavior of the original two–cell WTA circuit is shown in Figure 24. The graph shows (from top to bottom): currents flowing into the

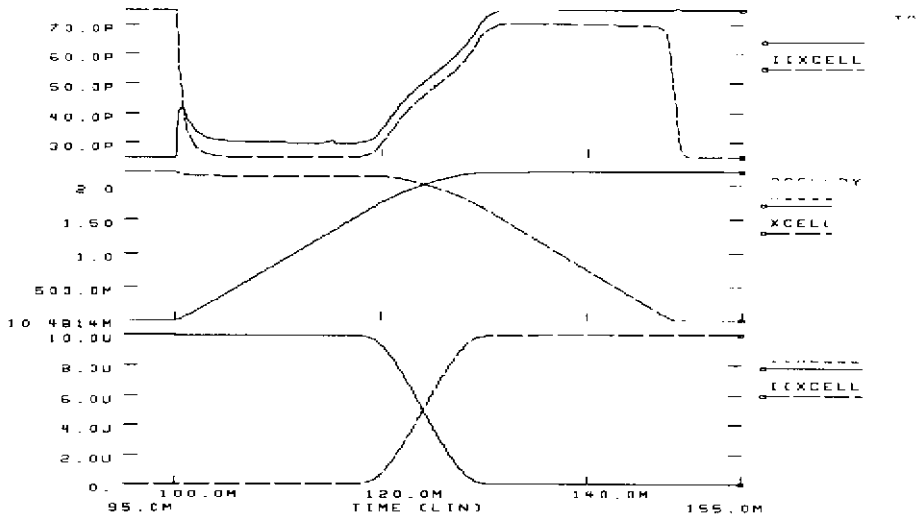


Figure 24: Two–cell WTA dynamic behavior. Graphs from top to bottom are: currents through input transistors T_{21} and T_{22}, voltages V_1 and V_2, output currents J_1 and J_2. The output is delayed by $T_D = 24ms$.

WTA's input transistors T_{21} and T_{22}, voltages V_1 and V_2, output currents J_1

and J_2. In the top graph we see that the currents flowing into the input transistors do not follow the transitions of the photocurrents. This is because initially most of the photocurrent flows into the parasitic capacitance C for changing the voltages V_1 and V_2. Therefore, the circuit is slow to respond to the input transition: the output current transition lags input for about *24ms*.

To improve the dynamic performance of the WTA circuit several measures can be taken: (1) increase photocurrent, (2) decrease parasitic capacitance C, and (3) reduce the voltage swing on the capacitance C. A modified WTA cell that implements all of these three measure is shown in Figure 25.

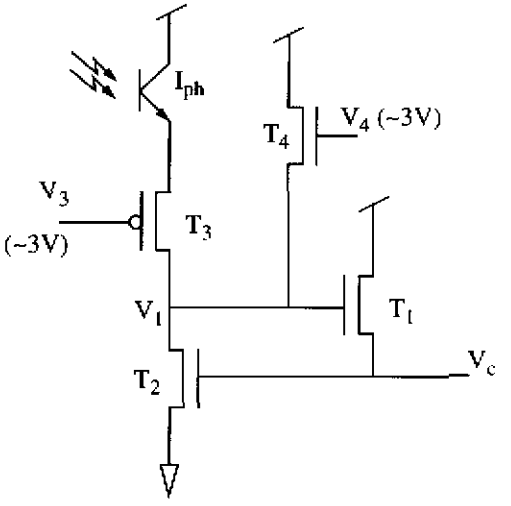


Figure 25: WTA cell with improved dynamic performance. (Shaded area indicates addition to the original WTA cell.)

Using a photo transistor instead of a photodiode gives a current gain of up to about *30*. The results of the simulation under the same input conditions as above are shown in Figure 26. The output current transition is delayed for *20ms*. The delay time improved, but not as much as one would expect. This is because the parasitic capacitance of the photo transistor is larger than that of the equivalent area photodiode.

By introducing the transistor T_3 as a current buffer, the voltage variations at the emitter of the photo transistor are greatly reduced. Consequently, the cumulative capacitance C is reduced, and the dynamic performance improved. The results are shown in Figure 27. The current buffer isolates the photodetector from the large variations of *V1* and *V2*. The bottom graph in Figure 27 shows the voltage variations on the photodetector. The variations are reduced from *2.5V* to about *10mV*. The photocurrent quickly

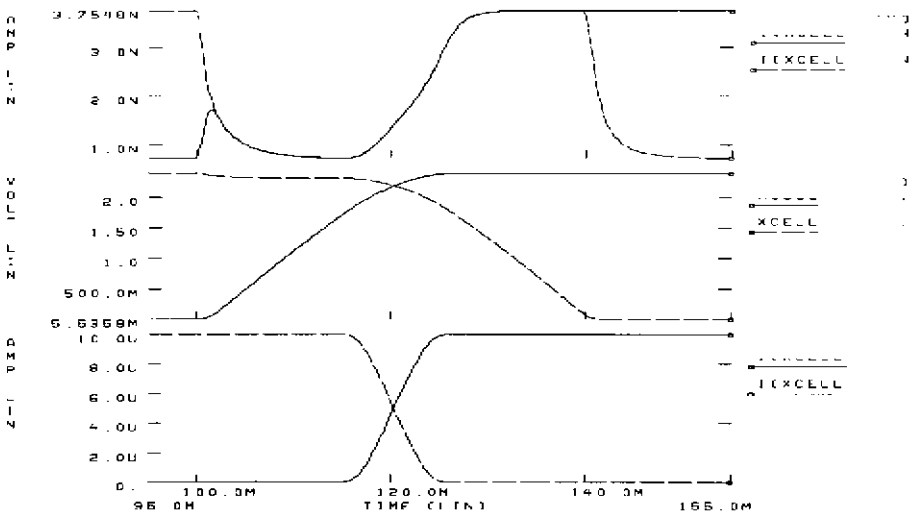


Figure 26: Dynamic response of the WTA employing photo transistors. $T_D = 20ms$

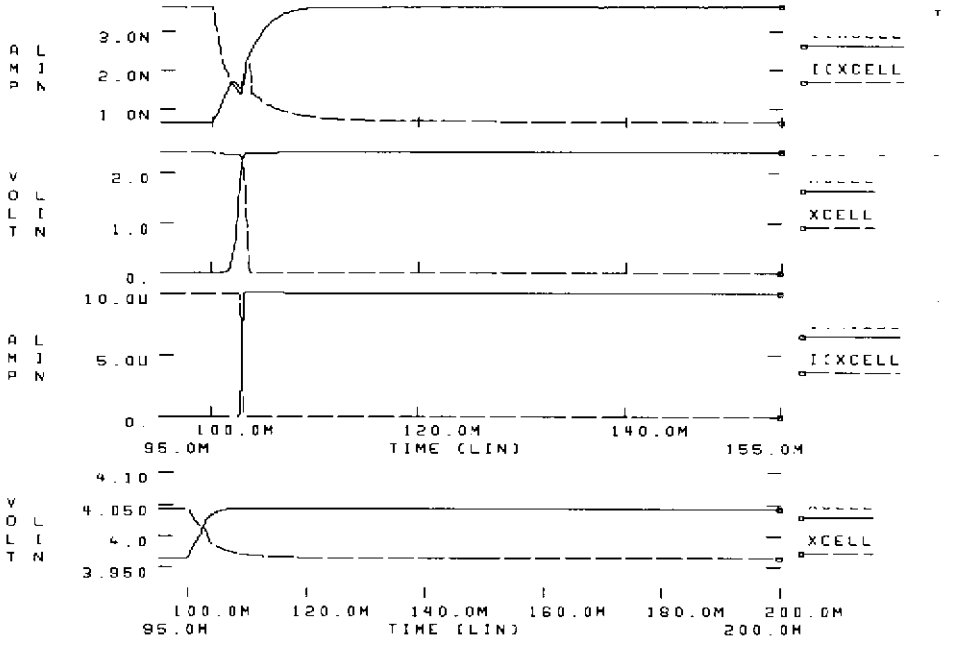


Figure 27: Dynamics of the WTA circuit employing the photo transistor and the current buffer. $T_D = 3ms$.

charges this difference and becomes available to the remaining circuitry.

The delay time is dramatically reduced from 20ms to about 3ms.

The remaining delay comes from the parasitic capacitance formed by the drain of the input transistor as well as the gait of the output transistor. These parasitic capacitances cannot be as easily isolated. However, we notice that the voltages V_1 and V_2 needlessly swing all the way down to the ground — a clear winner in the output currents is obtained as soon as one of the voltages goes to about $0.7V$ below the other one. This observation have led to the transistor T_4 (see Figure 25) which acts as a weak pull–up. Namely, as soon as the voltage V_1 or V_2 drop sufficiently bellow the bias voltage V_4, the T_4 starts to pull up preventing voltages V_1 and V_2 to drop bellow approximately $(V_4 - 0.7)$ volts. Figure 28 shows the dynamics of the two –cell

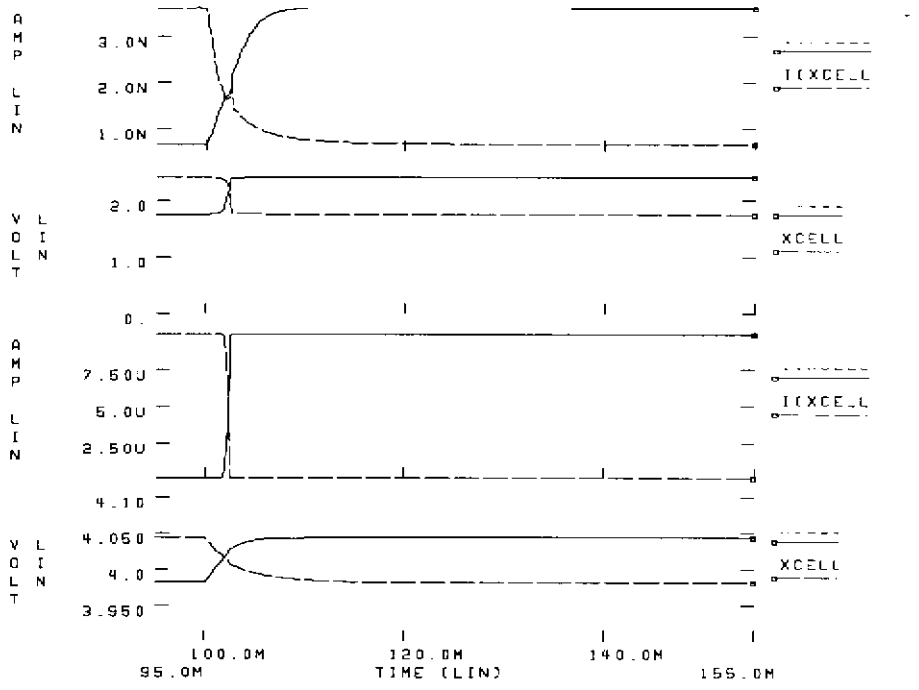


Figure 28: Dynamics of the WTA circuit employing the photo transistor, current buffering and voltage swing limitation. $T_D = 2.1ms$

WTA circuit employing photo transistor, current buffering and the pull–up transistor for voltage swing limitation. Under these conditions the delay time is further improved to about $2ms$.

6.2 *Attention Shifts*

In order to accommodate the top–down control of attention shifts suggested in Chapter 5, the sensor defines an active region. The active region — a subset of the entire array — can be programed by means of a row–column addressing. Only cells within the active region participate in the competition for the winner. This way the sensor directs attention to an interesting part of the retinal image.

6.2.1 Cell Inhibition

The active region is programmed by inhibition particular WTA cells under the external control. The inhibition of a cell is achieved by preventing the photocurrent from flowing into the input transistor. A circuit diagram of the WTA cell which implements this mechanism of inhibitions is shown in Figure 29. The shunting path for the photocurrent is provided through the

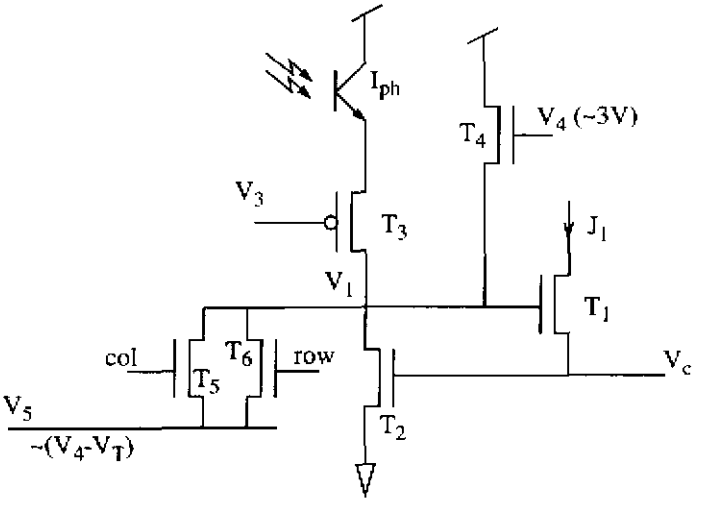


Figure 29: WTA cell with inhibition. (Shadowed area indicates components for cell inhibition.)

transistors T_5 and T_6. To maintain the cell active both $\overline{col}$ and $\overline{row}$ signals must be asserted (i.e. must be zero). This inhibition mechanism can be interpreted in two ways. One way is to say that when the photocurrent is shunted the cell effectively "sees" zero current and cannot win. The other way is to say that the switches T_5 and T_6 clamp the gate of T_1 to V_5, thus preventing T_1 from conducting current. The voltage V_5 must not be zero; it is about

$0.7V$ below V_4. This is consistent with the goal of minimizing the swing of V_1 and V_2 for improved dynamic performance. In addition, this ensures that when inhibited the T_1 is barely turned off and can quickly become enabled when needed.

6.2.2 Control of the Active Region

The appropriate row–column inhibition control defines the active region. This control is achieved from the periphery of the two–dimensional WTA array. The peripheral logic across three columns is shown in Figure 30. Similar logic is implemented for row addressing. For a cell within the WTA array to be active, both $\overline{col} = 0$ and $\overline{row} = 0$. In the select mode the active column band is programed by the content of the shift register. There are no restrictions on the width or location of the band, as any bit pattern can be entered into the shift register.

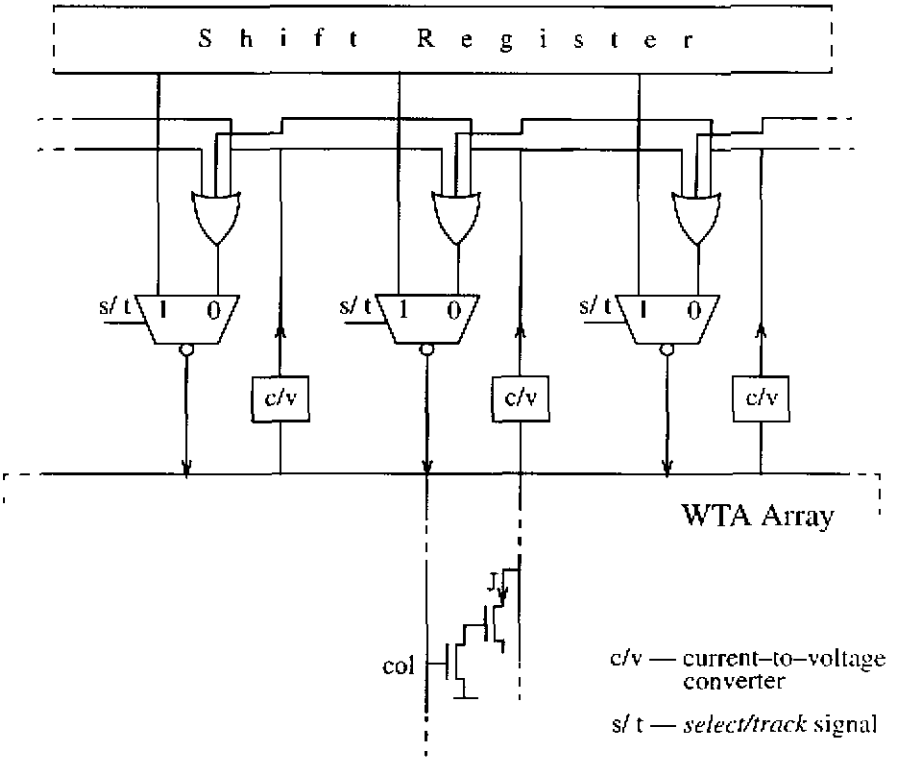


Figure 30: Peripheral logic for central control of the active region. The shaded area indicates one column. Similar logic is used for row addressing (not shown).

In the track mode the active region is programmed by the WTA array and is dependent on the location of the feature being tracked. A particular column is enabled if the winning feature is on that column, or on one of the two immediate neighbors. In the conjunction with the row inhibition (not shown), the tracking mode programs a 3 x 3 active region centered on the most recent feature. If that feature starts moving, one of the eight active neighbors will receive the winning feature and automatically update the position of the 3x3 active region. It is now clear that the salient feature is not necessarily the absolute maximum, but it is a local peak in the retinal image. If for any reason the tracking mode starts on a location which is not a local peak, the 3x3 active region will "slide" along the intensity gradient until it locks onto a nearby peak.

With moving objects the feature which is being tracked may reach the sensors edge and fall out of the field of view. In order to insure coherent behavior in these situations, the logic shown in Figure 31 is implemented.

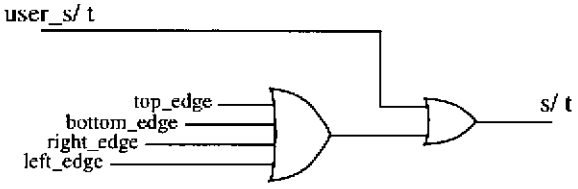


Figure 31: Logic for automatic switching between *select* and *track* modes.

The user may define the select mode by asserting signal *user_s/t*. However, when the user enables the tracking mode, the active region will be of size 3 by 3 as long as the tracked feature is not on one of the four edges of the array. When the feature reaches one of the four edges, the sensor automatically goes to a select mode. For a moment, the active region specified in the shift registers is enabled, and the absolute maximum is selected therein. Since the newly selected feature is no longer on the edge, the sensor automatically goes back to the tracking mode, shrinks the active region to a 3 by 3 size, and continues feature tracking.

6.3 *Experimental Data*

Two tracking sensors prototypes — 1D and 2D — have been built and tested for static and dynamic performance.

6.3.1 Static performance

The static performance has been tested on an early 1D prototype with 20 cells has been fabricated in 2μ CMOS technology. A cell occupies 40 by 59 microns. A photodiode uses about 60% of the cells area. The block diagram of the 1D prototype is shown in Figure 32.

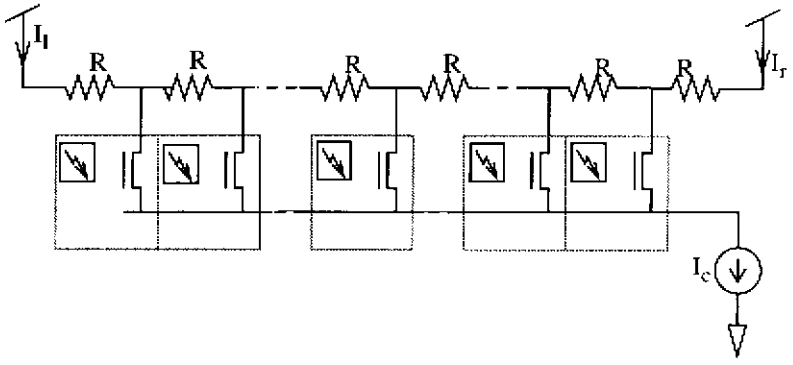


Figure 32: One dimensional optical position detector.

In the experimental setup, the chip was mounted behind a 25mm TV lens and oriented so that the array of cells ran horizontally. In one experiment, a dark plate (black photographic paper) with a bright vertical line on it (i.e. stretched white wire ⌀=0.5mm) was mounted on a calibrated translational stage. The stage was placed at about 520mm from the lens and oriented so that its line of motion was horizontal and transversal to the optical axes. For this arrangement, the field of view of a single photodetector was about 0.8 mm. Including the blurring effects of the lens we could say that the image of the target (i.e. wire) was smaller or comparable to the pixel size. This finding insures that only one photodetector or its immediate neighbors received an appreciable amount of light at a time. Therefore, unambiguous conclusions can be made as to whether a correct cell wins. The scene was illuminated with a 200W standard light bulb from a distance of about 3 meters behind the sensor and 1.5m above it. Therefore, illumination is considered uniform over the sensors field of view.

The target is moved horizontally throughout the entire field of view (i.e. 16mm) in steps of 0.2mm. The winning cell location (i.e. position) reported by the sensor is measured. The results are graphed in Figure 33. Also

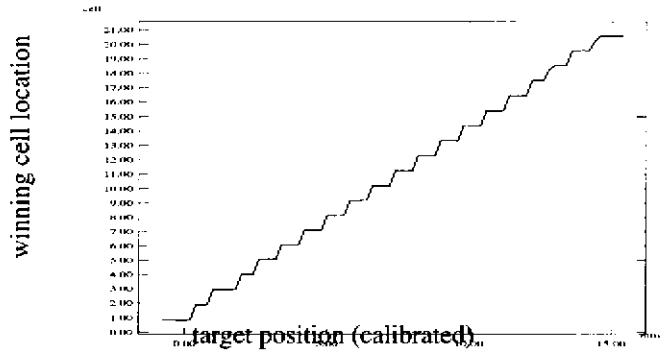


Figure 33: Winning cell localization reported by the 1D WTA computational sensor

measured was the voltage on the common wire. It is approximately logarithmically proportional to the intensity of the winning input photocurrent. Using computer simulation, the measured common voltage is converted to apparent input photocurrent (Figure 34).

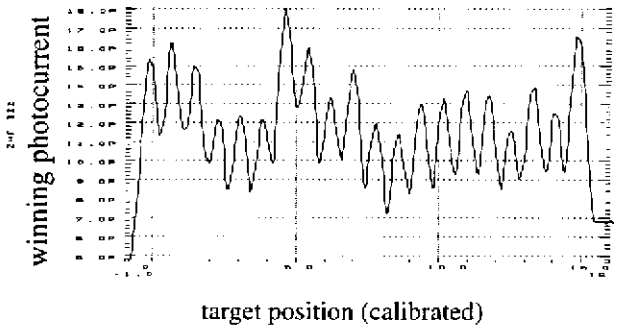


Figure 34: Apparent winning photocurrent.

In Figure 33 it is seen that the cells winning behavior as well as the winner localization is reported as expected. Namely, a particular cell remains a winner as long as the main portions of the bright target are focused on it. In the graph this is seen as the staircase line. As the target is moved its image leaves one cell and begins contributing photocurrent to the next one. At some point, the cell acquiring the target wins and takes control of the common voltage. As the arriving target moves towards the center of a

winning cell the intensity of the winning input current increases. The cell continues to win as the target passes the center, but its input current diminishes. In the mean time, the next cell begins to receive an increasing amount of light and the process continues. Therefore, as the target passes over the winning cell the measured common voltage increases, peaks, and then decreases. This behavior is clearly observed in Figure 34. The spacing of the peaks in the environmental coordinates is 0.79mm which, for the given experimental set up, matches the pitch of the cells.

Another important observation can be made from Figure 34. Even though a target of a constant intensity is scanned over the sensor, it doesn't result in equal peaks of common voltage. This is due to the device matching problem: the same target can be seen as one whose intensity apparently varies. A maximum peak of 19pA is observed at 4.5mm, while a minimum peak of 11.5pA at about 8.5mm. Also observed is a periodic pattern in peak variation over the chip area. This is due to the striation effects. The period as well as its relative amplitude of this variation are in a good agreement with findings in [Andreou et al. 1992].

The ratio of the two extreme perceived currents in this case is about 1.6. In the worst case, this means that if this circuit is to always correctly identify the winner, the strongest spot in the image must be about 1.6 times higher then the background. Transistor mismatch factors of 2 are typical for a MOS process [Mead 1989]. This imposes serious constraints on the nature of the input image, especially in the select mode of the tracking sensor. However, the devices within a small neighborhood match better, within 20% of each other. When the sensor is in the tracking mode only a small neighborhood is active. This means that in the tracking mode it is sufficient if the local image peak is only about 20% above the immediate neighbors. This is another example of benefits of top–down sensory adaptation.

6.3.2 Dynamic Performance

The dynamic performances is evaluated for a 28 by 28 cell two–dimensional tracking computational sensor. Each cell is 62μ square. The photo transistor takes about 30% of the cell's area.

The dynamic performance is evaluated using the experimental set–up shown in Figure 35. A scanning mirror projects a beam of light onto a white cardboard. This produces a dot which travels along a straight line. The sensor images the scene and tracks the moving dot. The rows of the sensor are aligned with the trajectory of the laser dot. Consequently, as the dot

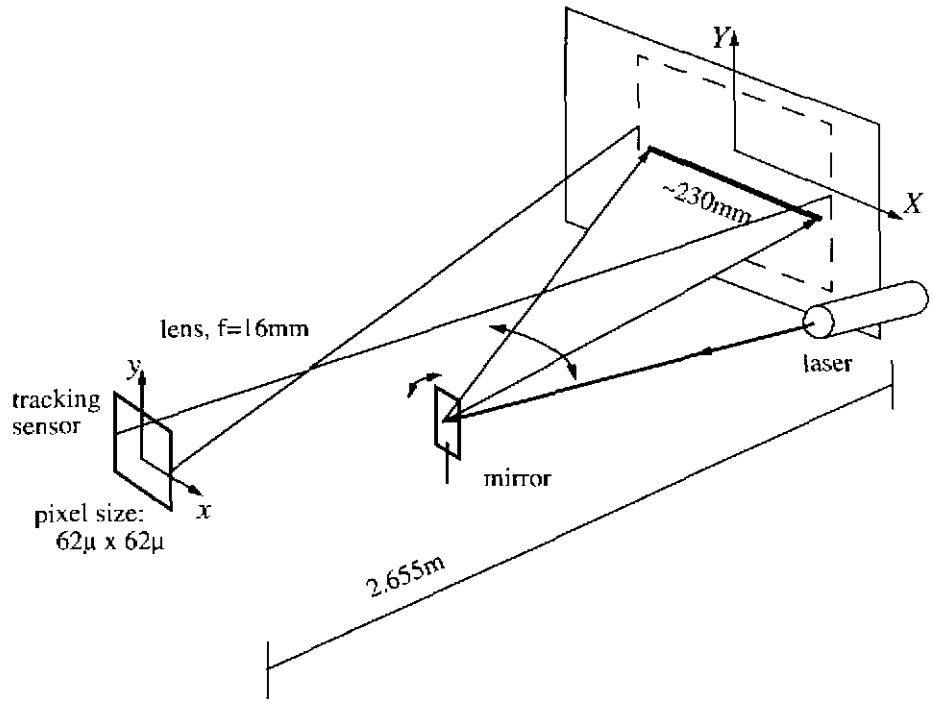


Figure 35: Experimental set up for evaluating dynamic performance of the tracking computational sensor.

travels from side to side only x position needs to be observed, since y position is not changing.

The mirror is driven from a sinusoidal oscillator whose frequency is adjustable. For the given setup the trajectory of the dot in the image coordinates is:

$$x(t) = A\sin 2\pi ft \quad [\text{cells}] \tag{6}$$

where A is the amplitude equal to 11.11 pixels, and f the scanning frequency. The instantaneous velocity in image coordinates is:

$$\dot{x}(t) = 2\pi fA\cos 2\pi ft \quad \left[\frac{\text{cell}}{\text{s}}\right] \tag{7}$$

with maximum being attained at the middle of the trajectory:

$$\dot{x}_{\max} = 2\pi Af = 69.8\,f \quad \left[\frac{\text{cells}}{s}\right] \tag{8}$$

The goal is to observe how quickly the tracking sensor can shift attention; that is, how quickly it can update the feature's location as the feature travels across the array of cells.

Influence of the Current Buffer and the Pull–up

The first set of test is performed to show contribution of the current buffer T_3 and the pull–up transistor T_4. The effects of the current buffer and the pull–up can be turned on or off by biasing V_3 and V_4 respectively.

Without the buffer and the pull–up the sensor was reliably tracking up to the scanning frequency of 33Hz or 2,303.6 cells per second. Figure 36 shows

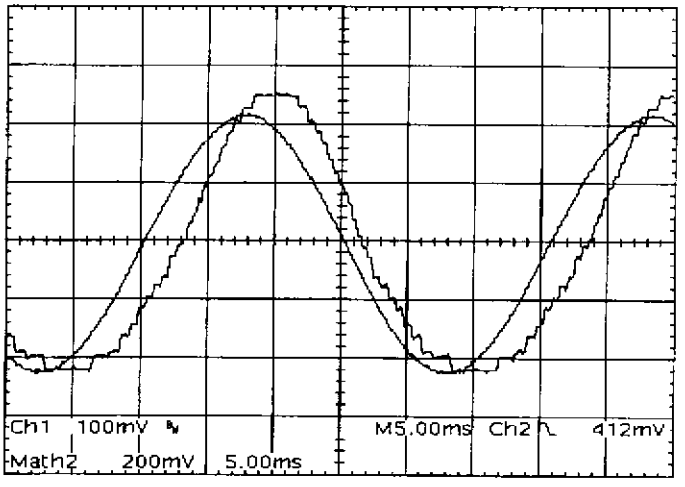


Figure 36: Tracking performance without the current buffer and without the pull–up. f=33Hz.

two measured waveforms: (1) the feature's position x as reported by the tracking sensor, and (2) the sinusoid driving the mirror. If the frequency of the mirror is further increased, the reported position begins to distort. This is illustrated in Figure 36 for the scanning frequency of 83Hz. As expected the tracking capability of the sensor starts to break down in the middle of the trajectory as the velocity of the feature is the greatest there.

In the next experiment the current buffer is turned on by biasing V_3. As expected the dynamic performance improved: the maximum tracking frequency is increased from 33Hz to about 83.3Hz or from 2303.6 to 5793.9 cells per second. This is shown in Figure 36: previously distorted waveform for the feature's position is now better resembling the sinusoid.

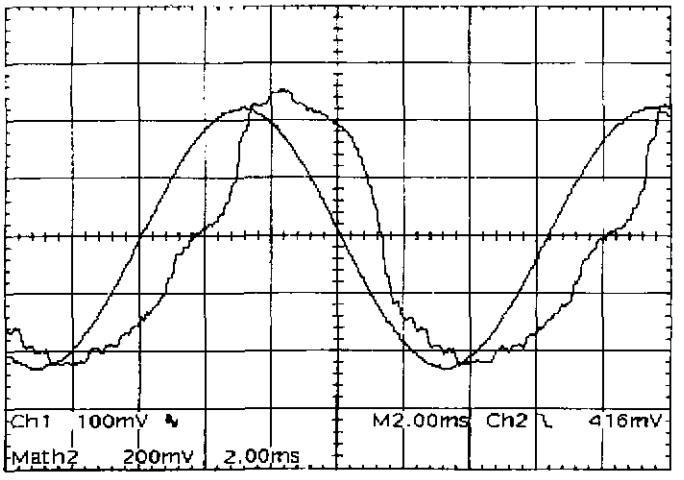


Figure 37: Tracking performance without the current buffer and without the pull–up. f=83Hz.

Finally, the pull up transistor is turned on by biasing V_4. The dynamic performance is slightly improved as showed in Figure 36 — the feature tracking is improved from 83Hz to about 100Hz, or 6980.6 cells per second.

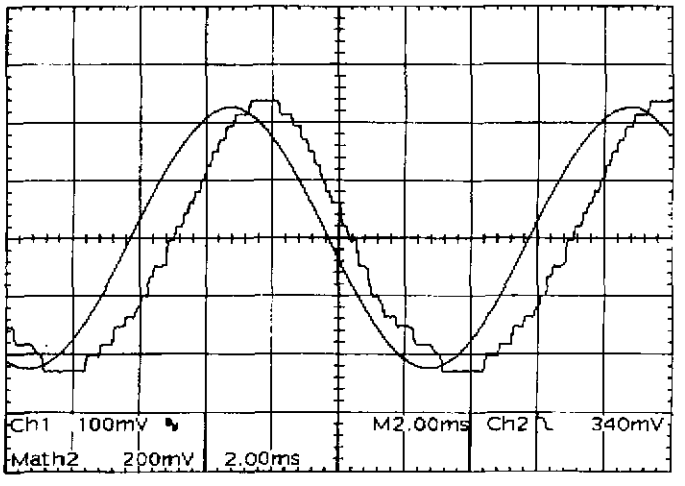


Figure 38: Tracking performance with the current buffer but without the pull–up. f=83Hz.

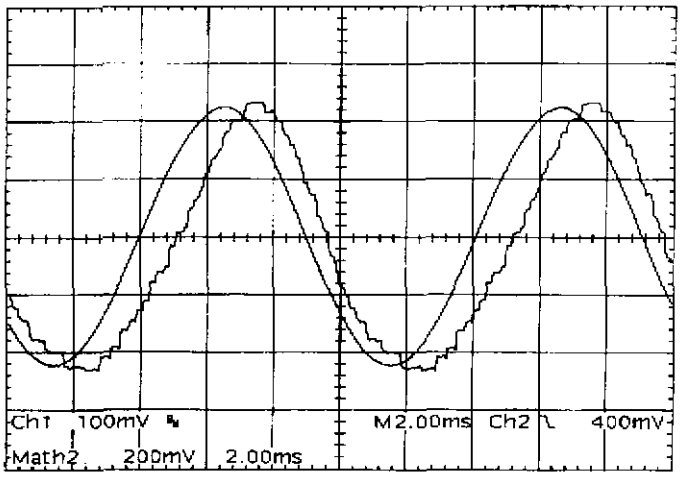


Figure 39: Tracking performance with both the current buffer and pull–up. f=100Hz.

The dynamic performance of the WTA is summarized in .

Summary of the experimental findings for the winning/loosing dynamic performance for the WTA circuit.

Basic WTA	WTA with the buffer	WTA with the buffer and pull–up
2303.6 cells/s	5793.91 cell/s	6980.6 cells/s

Influence of the Features Intensity

Another set of experiments is performed to evaluate how the intensity of the feature influences the dynamic performance. Using neutral density filters placed in front of the sensor's lens, the light is controllably attenuated. For each filter the frequency of the mirror is increased until the waveform of the feature's position begins to distort. This way the maximum frequency is estimated for each intensity.

Two sets of experiments are performed: (1) without the buffer and the pull–up, and (2) with the buffer and the pull–up. The results are graphed in Figure 40.

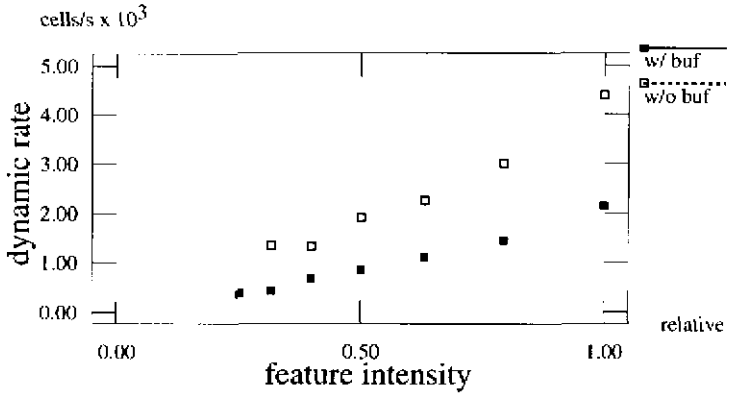


Figure 40: Maximum angular velocity of the attention shifts as a functions of the relative feature intensity.

6.4 Applications in Robotics

The tracking computational sensor is an intelligent position sensitive detector: it selects a feature and tracks it by ignoring the background. There has been considerable work done on finding the position of a bright object using digital means. Special purpose systems have been developed for tracking objects, and others, to compute object moments. Most of these systems are not "seeing" chips and require serial input of the binary images. These systems update object parameters at the standard video rate of 30 times per second with at least 1/30th of a second latency.

An analog seeing chip that computes the centroid of a bright object focused on the chip, has been built by DeWeerth and Mead at Caltec [18]. This system computes the centroid of the intensity distribution focussed over the entire chip. For a meaningful performance, the chip requires a bright object on a low intensity background. The centroid is calculated using the projection method, and the gradient descent method to compute the centroid of each projection distribution.

Hamamatsu company manufactures "position sensitive detectors". They are a single large square photodiode [29]. One of the diodes layers is used as a continuous resistive sheet which functions as a current divider. Portions of the photo–currents are monitored at the four edges of the diode and are indicator of the centroid of the intensity distribution focused on the diode. This detector exhibits significant spatial non-linearity and, like the previous

example, requires a bright object on a dark background. Nevertheless, Hamamatsu's PSDs are in practical use in many systems.

Standley and Horn at MIT have built a focal–plane analog chip for position and orientation computation [66]. An image is focused on the chip, thresholding is performed at each cell, and the result is injected into a 2D resistive grid of polysilicon resistors.The periphery of the resistive grid is grounded and currents flowing into the ground are monitored. It can be shown, using Green's theorem, that these peripheral currents contain enough information to infer position and orientation of an object. A 29 by 29 cell detector is built on a 7.9 by 9.2 millimeter chip. This is an inferior resolution compared with the cell size of the tracking computational sensor.

Active Triangulation Range Sensing

Triangulation is a range finding technique employing a structured illumination and an imaging sensor [8]. In the light–stripping triangulation, a plane of light is either discreetly moved or continuously swept across a scene. For a given position of the light plane, an imager sees a bright line revealing a local profile of the scene. The distance to the scene is inferred from the *position* of the illuminated pixel and the geometry of the optical setup.

Triangulation employing a plane of light is in essence the row–parallel technique, i.e. the position of an illuminated pixel in each *row* is relevant information. This information is easily obtained with the tracking computational sensor. Suppose that a 1 by N active region is scanned from top to bottom. Such an active region observes one row at a time and determines the position of the laser stripe within that row. Then the active region selects next row and reports the position therein. If the sweeping of the laser is slower than the scanning of the active region, by repetitive scanning from top to bottom a two–dimensional range map of the observed environment is collected. The scanning of the rows is limited by the dynamics of the tracking sensor. There are faster sensors that operate on a similar principle [25] [26]. Nonetheless, this application of the tracking computational sensory illustrates two things: (1) its versatility in a range of practical applications, and (2) a feature saliency can be achieved in intensity images, through clever image formation.

Active Illumination Stereo

In the case of the triangulation, the knowledge of the position of the plane or line of light is essential. This is not easily and accurately obtained in prac-

tice. To remove the requirement for an accurate light beam position estimate a second tracking sensor can be introduced to form a stereoscopic pair with the first one. A scene can be illuminated with a scanning laser *beam*. In essence, this is a stereo vision system with structured light illumination: *structured light stereo.*

The structured light eliminates the well known problem of correspondence associated with arbitrary image stereo. With a laser beam, each of the sensors sees only one bright spot in the entire field–of–view. This is the feature in the scene that has trivial correspondence in the two images. The position of such a feature is readily detected by the two WTA sensors. The output quantities of the sensors are (x_l, y_l) and (x_r, y_r), i.e. the feature position in the left and right sensor, respectively. Disparity is found as Euclidean distance between these two points:

$$d = \sqrt{(x_l - x_r)^2 + (y_l - y_r)^2} \tag{9}$$

It is possible, however, to arrange the two sensors in such a way that the epipolar lines are parallel to the rows of the sensor arrays. In such a case, computing disparity simplifies to:

$$d = x_l - x_r \tag{10}$$

From a computational point of view this is a desirable arrangement. In practice, it is easily done by mechanical calibration of the relative position of the sensors until the $y_l \approx y_r$ condition is always observed.

In addition to the position of the feature, the WTA network provides information about the intensity of its winning input photocurrent. These values are labeled as e_l and e_r, and are available on the common wire of the left and right sensor respectively. They encode the intensity of the light received after it has reflected off the scene. Therefore, this is the intensity image of the scene.

Due to the fact that the two sensors are viewing a scene from two different view points, the object sometimes occludes a certain portion of the scene to one sensor but not to the other. This gives rise to the problem of occlusion: a feature (i.e. the illuminated spot) may be seen in one image but not in the other. Occlusion has been a stumbling block in many stereo vision algorithms. With our proposed system this situation is easily detected. Since each sensor provides information about the intensity of a detected feature on the common wire. If the features are images of a same illuminated bright spot in the scene, the left and right sensors report the similar intensities.

However, if occlusion occurs the two values differ significantly which signals the problem.

Motion Sensing and Visual Servoing

The tracking sensor continuously detects the position of the feature. The temporal measurement of the feature's position can be used in visual servo systems. In addition, time derivatives of the x and y position yield velocities $\dot{x}$ and $\dot{y}$ respectively. In addition, the least significant bit (LSB) signal of the digitally encoded position of the feature, has frequency that is proportional of the velocity of the feature. In either case, the sensor could be a simple motion/velocity detector.

Imaging

Occasionally the tracking computational sensor can be scanned for image readout. Due to the device mismatch the sensor has highly nonuniform field, and images are noisy. Nonetheless, the ability to readout even noisy images can be useful in many embedded applications. One can imagine a scenario in which the sensory track features, but occasionally the supervising processor may want to obtain the image and define a new active region within which the sensor should continue selecting features for tracking.

Chapter 7

Intensity–to–Time Processing Paradigm

The dimension of time plays a dominant role in neural processing. The neural responses are digital in amplitude but analog in time; therefore, it has been suggested that time is a core factor in the flow and transformation of information in the brain [6].

The *intensity–to–time processing paradigm* is an efficient solution for massively parallel global computation over large groups of fine–grained data [12]. Inspired by biological vision, the intensity–to–time processing is based on the notion that stronger signals elicit responses before weaker ones. Assuming that the inputs have different intensities, the responses are separated in time and a global processor makes decisions based only on a few inputs at a time. The more time allowed, the more responses are received; thus, the global processor incrementally builds a global decision first based on several, and eventually based on all the inputs. The key is that some preliminary decisions about the retinal image can be made as soon as

the first responses are received. Therefore, this paradigm has important place in low–latency vision processing.

7.1 Latent Period of Biological Vision

The latent period of vision is the time interval elapsing from the moment a light stimulus is absorbed to the onset of the physiological response to the stimulus. As experiments on the human visual system involve introspection, the most that can be achieved is the measurement of the reaction time; the psychological experiment evaluating latency of vision cannot with certainty reveal where exactly this latency comes from. The following experiments and their results have been summarized in [61].

Arden and Weale used two flashing light stimuli, entering one eye each. Each test field subtended a few minutes of arc at the eye. The intensity of one of the stimuli was variable and relative phase between the two was adjustable. When the intensity and the relative phase of flashing lights were adjusted in such a way that both stimuli appeared to flash at the same time, a measure of the relationship between the latent period and the stimulus intensity was obtained. Latent period varies inversely with stimulus intensity and is more accentuated in the periphery than in the fovea (see Figure 41).

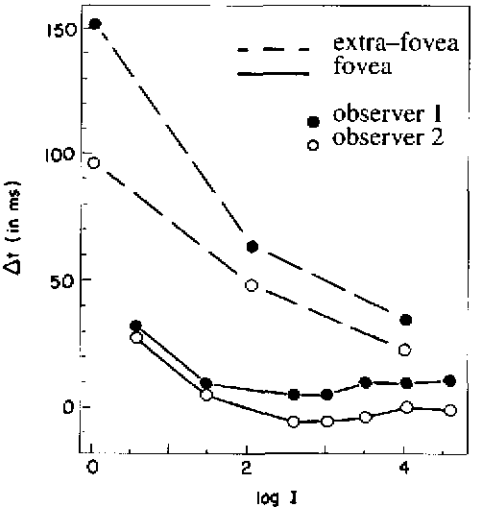


Figure 41: The change in the latent period of vision with the intensity of stimulus expressed as multiples of the absolute threshold of vision. (Measured by Arden and Weale in 1954) [61].

In a more extensive study, using a 1° field, Roufs showed that the change in the latent period is:

$$t - t_o = -T\log_e E/E_o \qquad (11)$$

where T is a constant equal to $10ms$ and E is the variable retinal illumination, the standard value E_o of which is associated with a latent period of t_o.

The latent period dependency upon the luminance variations also appears to underlie the stereo effect described in 1922 by Pulfrich [60]. When a swinging pendulum is viewed binocularly, and a neutral density filter is placed in front of one eye, the apparent trajectory of the pendulum bob follows an ellipse or circle, depending on the pendulum period, viewing distance, filter transparency and other factors. The explanation for this astonishing phenomenon is that the image of the filtered eye is delayed, because the lower intensity entails a longer latent period, thus causing the disparity between the left and right image. The brain receiving these asynchronous messages puts them together as if originating at the same instant, thus interpreting perceived image disparity as the depth of the bob.

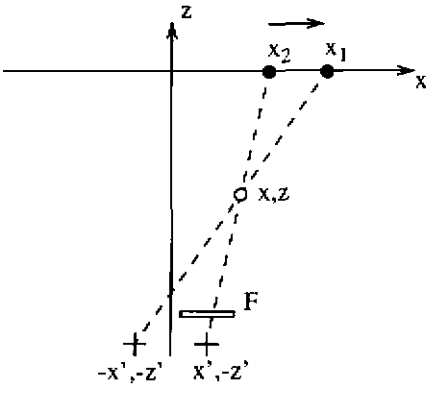


Figure 42: An explanation of the Pulfrich effect.

This explanation is illustrated in Figure 42. The two eyes are located at $-x', -z'$ and $x', -z'$. The pendulum swings approximately along the x–axis. Due to the neutral density filter F, the right eye operates at the longer latent periods than does the left one. Consequently, the free eye sees the pendulum bob at point x_1, while the filtered eye perceives it at point x_2. The brain fuses the two impressions and interprets that the bob is at x, z. As the viewing distance increases the perceived trajectory of the pendulum bob

goes from a flat ellipse to a circle, and for an even longer distance to an elongated ellipse (see Figure 43).

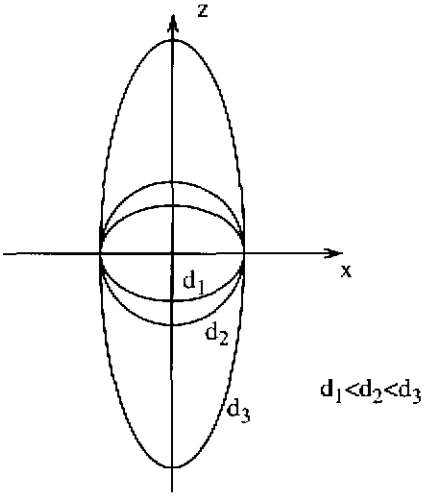


Figure 43: Apparent theoretical trajectory of pendulum bob when the viewing distance *d* is varied.

The above experiments measure the latent period by "cheating" the vision system: by controllably introducing artificial latency in the visual channels, the brain processing, trained for "natural" situations, is altered to perform different function (i.e. to perceive illusions). Conversely, one may hypothesize that the latency in the system plays a role when processing under normal conditions: the brain may utilize the temporal disparity of the individual responses for computation. In fact, using the luminance–to–latency relationship has been suggested for improving segmentations in computer image processing [13].

7.2 Proposed Implementation

Assuming that the inputs have different intensities, the responses are separated in time as a natural consequence of their intensity–to–time relationship. Considering the limited communication and processing capacity in machines this temporal separation is attractive for performing global operations over large groups of data: as the visual responses are separated in time, both the communication and global processing resources are protected from information overload. Furthermore, this concept is consistent with the goal of producing a fast–reacting vision system: as soon as the first

responses are received at the global processor, early conclusions can be made and an action initiated. Furthermore, these early conclusions can be used for the top–down sensory adaptation, thus aiding further information processing.

The intensity–to–time paradigm for massively–parallel fine–grain processing in computational sensors is implemented by the architecture shown in Figure 44 and involves the following. A large number of input data are gathered optically by focusing an image onto the array of sensor–processor cells. Each cell has a local processor which performs one or more predetermined (i.e. pre–wired or pre–programmed) operations. The instant when these operations are executed, or triggered, is determined by a photo–sensitive control element. The photo–sensitive control element receives light, and fires a response after a latent period which is inversely proportional to the intensity of the received light. There is also a global processor which receives and/or sends signals to the array of cells. Since local processors trigger at times determined by the magnitude of their input operands, the global processor serves only a few local processors at a time.

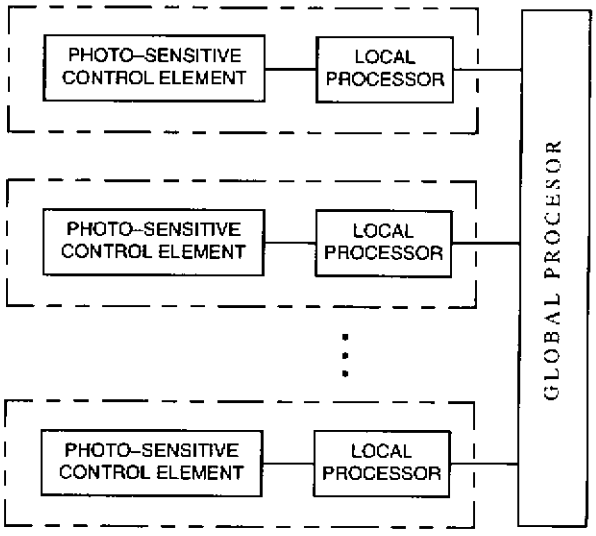


Figure 44: An architecture of computational sensor implementing the intensity–to–time processing paradigm.

The latent period can be implemented with a control element whose block diagram is shown in Figure 45. In this case, the latent period of the control

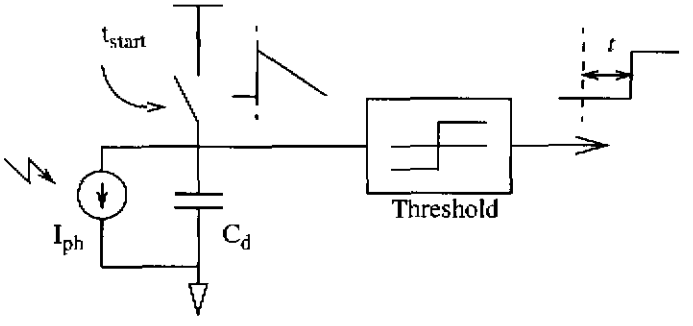


Figure 45: Radiation control element implementing the latent period.

element is inversely proportional to the received intensity:

$$t = \frac{C}{E} \tag{12}$$

where E is the intensity and C is a constant. The ratio of the latencies then becomes:

$$\frac{t_o}{t} = \frac{E}{E_o}$$
$$\log t - \log t_o = -\log \frac{E}{E_o} \tag{13}$$

where t_o is latency corresponding to a standard input intensity E_o. The relationship in Equation 13, resembles that of Equation 11 found in human vision.

By using the intensity–to–time paradigm we have developed a *sorting computational sensor* — an analog VLSI sensor which is able to sort all pixels of an input image by their intensities. Photon integration is initiated simultaneously for all the cells. The intensity–to–time paradigm orders inputs in time according to their magnitudes, with the strongest input responding first. At each given time the global processor maintains the analog count of the inputs that have responded. This count is stored as an index within a cell(s) which responds next. Once all the inputs have responded, the array contains the "image of indices". This sorting method is

closely related to an algorithm for sorting integers known as the counting sort [16].

The computed image of indices has a uniform histogram, i.e. all indices occur equally frequently (ideally once) in the image of indices. Such images have several useful properties: (1) the contrast is maximally enhanced, and (2) the dynamic range of the readout circuitry is equally utilized. The detailed implementation and applications of the sorting sensor are described in the next Chapter.

7.3 Similarity with Biology

There are several similarities between the intensity–to–time processing paradigm and biological neural processing.

Large number of inputs. Intensity–to–time takes large numbers of input, but uses only a subset of them at a time for building decisions.

Cooperative parallelism. If used with appropriate local and global processors, the intensity–to–time implements globally cooperative parallelism. For example, in the sorting sensors, the index of a particular local processor is derived from the global temporal observation of the entire array.

Precision vs. massive parallelism. The intensity–to–time uses massive parallelism rather then precision to deliver useful information. For example, in the sorting sensor with N cells, the available dynamic range of the output circuitry is used in N uniformly distributed "quantizations" levels. The output amplifier, therefore, transmits uniformly distributed, or equiprobable messages. From information theory it is known that the source that transmit equiprobable messages maximizes amount of information delivered to the receiver [63]. This means that the sorting sensor utilizes available bits in the most effective way. Furthermore, as the number of cells, N for imaging sensors tends to be large (>40,000), the massive number of cells are "fighting" over occupying the available dynamic range at the output. Individual cells might be noisy, but due to the massive parallelism this noise at the output might be suppressed bellow the noise margin of the output circuitry and the A/D converter.

Trigger feature. Like biological nerves, each photo–sensitive control element can be considered a firing neuron which responds to a single input stimuli. The responding signal which is digital in voltage, but analog in time. The stronger the input, the sooner the neuron fires.

Signal transmission. The intensity–to–time separates input responses in time. The individual responses are infrequent: they happen only once during each frame. In biological system, each such response would have its own axon. The axon would transmit this signal when it happens but would remain underutilized during other times. The intensity–to–time paradigm takes advantage of the fact that the responses are infrequent and separated in time, and uses a single wire to transmits all the responses to the global processor. This is a departure from biological model. It shows that for an appropriate input signal representation, the available speed in VLSI, can be used in place of fully connected biological wiring [50] [55].

Chapter 8

Sorting Computational Sensor

By using the intensity–to–time paradigm we have developed a sorting computational sensor — an analog VLSI sensor which is able to sort all pixels of an input image by their intensities, as the image is being sensed.

8.1 Circuitry and Operation

Shown in Figure 46, the sorting sensor is comprised of a sensor–processor cell array and a global processor. A resistor R, a voltage buffer and wires W_{in} and W_{out} comprise the global processor. The global processor communicates with the array of sensor–processor cells over the wires W_{in} and W_{out}. Each cell has a local processor and a photo sensitive control element. The local processor is comprised of a track–and–hold (T/H) circuit, and a current signal generator. Accordingly, the local processors perform two functions: (1) data supplied by the global processor via W_{in} are memorized in the T/H

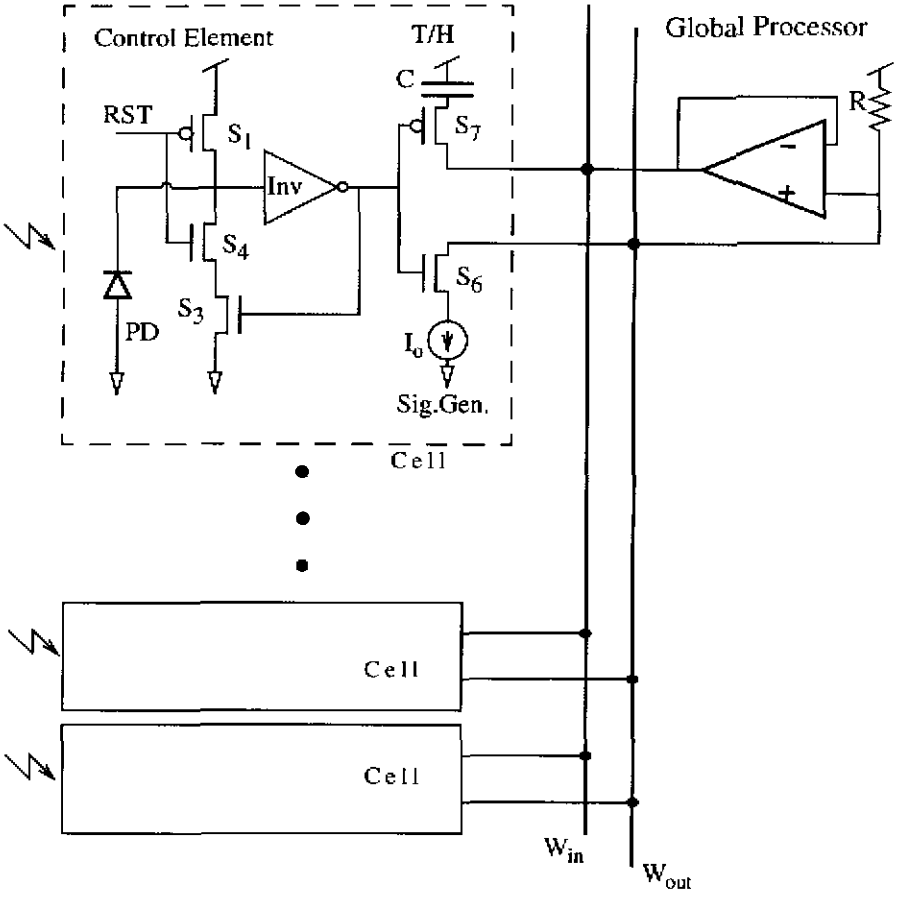


Figure 46: Schematic diagram of the sorting computational sensor

circuits, and (2) a current signals I_o is sent to the global processor via W_{out}. The remaining portion of the cell comprises the photo sensitive control element, which controls the instant when the corresponding local processor executes its functions.

Figure 47 shows the simulation of the circuit operation for the sorting sensor with four cells. A photodiode (*PD*) operating in the photon flux integrating mode [71] detects the light. In this mode of operation the capacitance of the diode is charged to a high potential and left to float. Since the diode capacitance is discharged by the photocurrent, the voltage decreases approximately linearly at a rate proportional to the amount of light impinging on the diode (Figure 47, top graph).

The diode voltage is monitored by a CMOS inverter (*Inv*). Once the diode voltage falls to the threshold of the inverter, the inverter's output changes state from low to high (Figure 47, second graph). A switch S_3 is included to force rapid latching action.

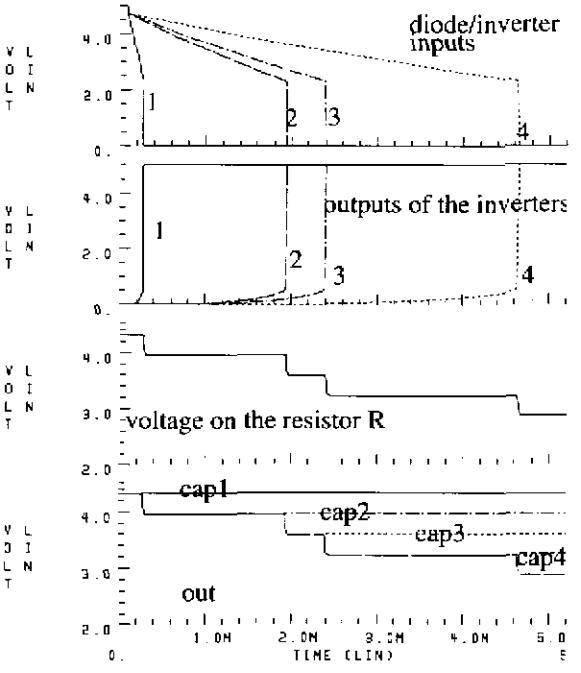


Figure 47: Sorting computational sensor: a four cell simulated operation.

The output of the inverter represents a control signal produced by the photo sensitive control element. The control signal determines the *instant* when the capacitor C in the T/H memorizes the signal from the wire W_{in}, as well as when the current I_o is supplied to the wire W_{out}. This is achieved by two complementary switches: S_7 disconnects the storage capacitor C from the global input wire W_{in}, and S_6 turns on the internal current source I_o.

Currents from all cells are added together in the output wire W_{out}; therefore, this wire functions as a global adder. The current in the output wire continuously indicate the count of cells that have responded. This current is converted into voltage via the resistor R. The third graph in Figure 47 shows this voltage. It represents the index of a cell that is changing state and is supplied to the global input wire for storage within the appropriate cell(s). The capacitor within each cell follows this voltage until it is disconnected.

At that point a capacitor C retains the index of the cell (Figure 47, bottom graph). The cell with the highest intensity input has received the highest "index", the next cell one "index" lower, and so on.

The sorting sensor computes several important properties about the image focused thereon. First, the time when a cell triggers is approximately *inversely* proportional to the input radiation received. Second, by summing up the currents I_o from all the local processors the global processor knows continuously how many cells have responded. This time waveform is closely related to a cumulative histogram of the input image [8]. The time derivative of this signal is related to a histogram of the input image. This is one global property of the input image that is reported by the chip with very low latency.

8.2 VLSI Realization and Evaluation

A 21 x 26 cell sorting sensor has been built in 2μ CMOS technology. The size of each cell is 76μ by 90μ. The photodiode takes 13% of the total cell area. The micrograph of the chip is shown in Figure 48.

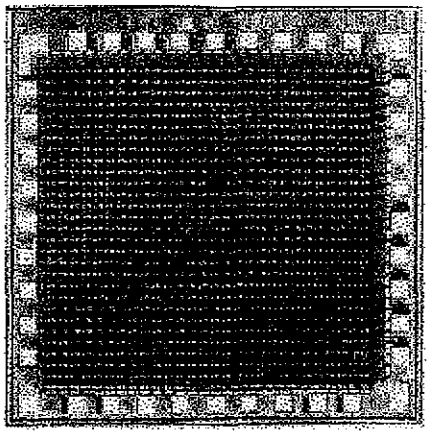


Figure 48: Micrograph of the sorting chip.

An image was focused directly onto the silicon. The cumulative histogram waveform, as well as the indices from the sorting sensor were digitized with 12 bit resolution. In order to facilitate a hard copy reproduction the 26x21 images obtained by the sorting chip are interpolated and magnified by the factor of 2.

Scene *1*, one setting in an office environment, was imaged by the sorting chip under common office illumination coming from the ceiling. Figure 49

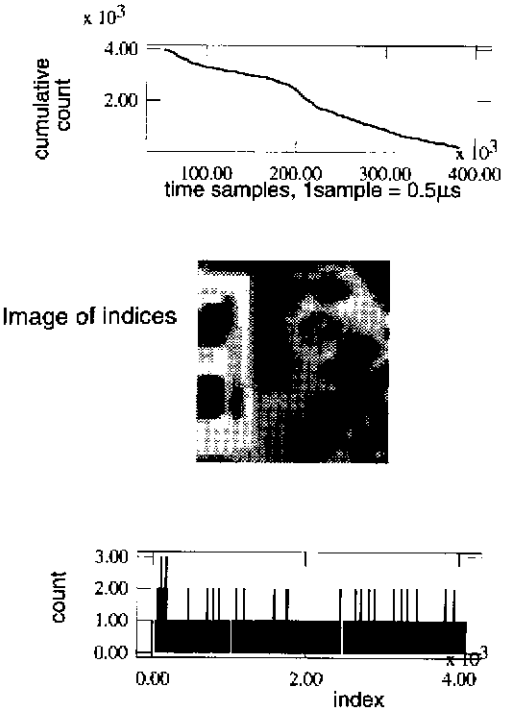


Figure 49: Scene 1 imaged by the sorting computational sensor.

shows the cumulative histogram of *the scene* and the image of indices both computed by the chip. The bottom graph in Figure 49 shows frequency of occurrence of indices. Most cells received different indices, as they detected different input intensities. Occasionally as many as 3 pixels were assigned the same index. Overall the histogram of indices is uniform, indicating that the sorting chip has performed correctly.

There is a total of 546 pixels in this prototype, and most of them received different indices. This means that without special considerations as to the illumination conditions, low–noise circuit design and temperature and dark current control, our lab prototype readily provided images with more then 9 bits of resolution. The range of indices (from 0 to 545) remains unchanged and the indices maintain uniform histogram regardless of the range of input light intensity or its histogram. Therefore, the sensor delivers the most information to the user since all the levels are equiprobable. This is the most optimal, or maximal, use of bits.

Another setting, Scene *2,* from the same office was also imaged. Figure 50 shows the scenes's cumulative histogram and image of indices, as well as the frequency of occurrence of those indices. Scene 2 (Figure 50) contains

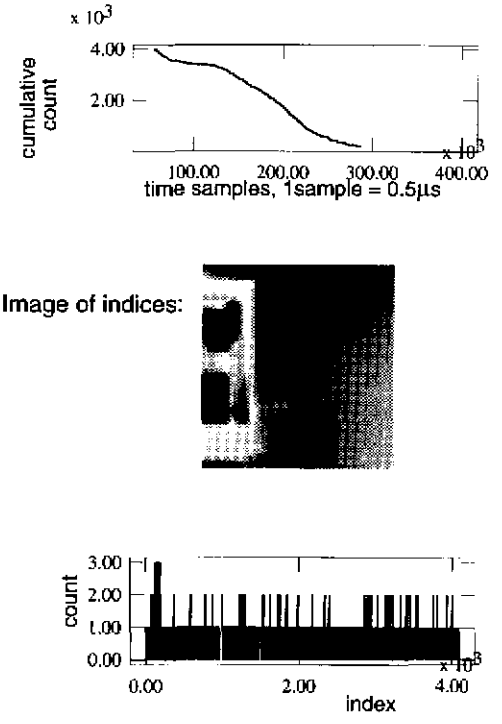


Figure 50: Scene 2 imaged by the sorting computational sensor.

more bright regions than Scene 1 (Figure 49) because the moderately dark regions of the person in a partial shadow are replace by the bright wall. Consequently, the sorting chip takes less time to compute Scene 2 than Scene 1. The total time shown on the time sample axis of the cumulative histograms is about *200ms.*

It is important to note that the performance of the sorting computational sensor readily scales–up with the image size. The frame rate is only related to the illumination of the scene, not to the size of the array. For well illuminated environment, the computation time is from 10–30ms. A higher resolution sorting computational sensor can be produced in more advanced VLSI technology. For example, in 1.2μ CMOS technology the pixel size is 38μ by 38μ. One such prototype is being fabricated as this thesis is being written.

8.3 Sorting Sensor Image Processing

The sorting computational sensor computes the cumulative histogram of the sensed environment. This global waveform is supplied and stored in the local processor of the array. Therefore, the cumulative histogram waveform defines custom mapping for each frame — a mapping from the input intensities to output indices. In general, this global waveform enables the sorting sensor to perform numerous other operations/mappings on input images. Examples of such operations include histogram computation and equalization, arbitrary point-to-point mapping, region segmentation and adaptive dynamic range imaging. In fact, in its native mode of operation — sorting — the chip provides all the information necessary to perform any mapping during the readout. In the following examples, the sorting sensor computes the cumulative histogram and image of indices and a particular mappings are then performed in software.

8.3.1 Histogram Equalization

When the voltage of the cumulative histogram (computed by the chip itself) is supplied to the local processors, the generated image is a histogram equalized version of the input image [8]. This is the basic mode of operation for the sorting chip and has been illustrated in the previous section.

8.3.2 Linear Imaging

When the waveform supplied to the input wire is inversely proportional to time, the values stored in the capacitors are proportional to the input intensity. This mapping implements a linear camera.

By producing the cumulative histogram waveform and the image of indices, the sorting computational sensor provides all the necessary information for the inverse mapping — the mapping from the indices to the input intensities. Figure 51a shows the image of indices for Scene 1 and the image of inferred input intensities. Figure 51b shows an image taken by a commercial CCD camera for showing natural light conditions in the office environment from which Scene 1 was taken. The inferred input intensities closely resemble the natural condition in the environment.

8.3.3 Scene Change Detection

Analyzing the change in the histogram pattern is a basic technique to classify images or detect a scene change. The sorting computational sensor

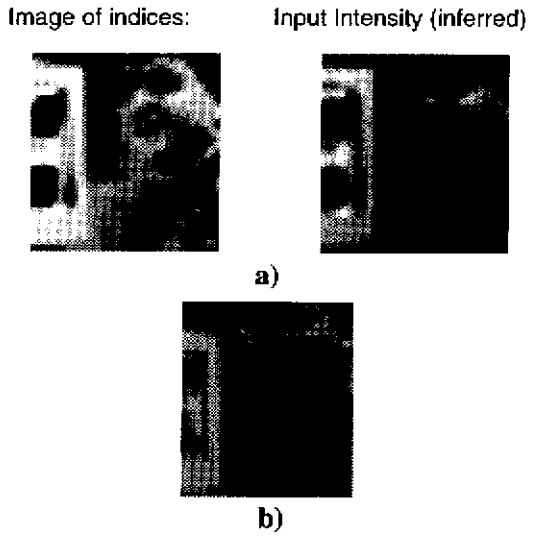


Figure 51: a) Indices from the sorting sensor and inferred input intensity, b) CCD camera image.

computes the cumulative histogram at real–time and can be used for low-latency scene discrimination/surveillance without requiring the image to be read out. The illustration of this technique is shown in Figure 52.

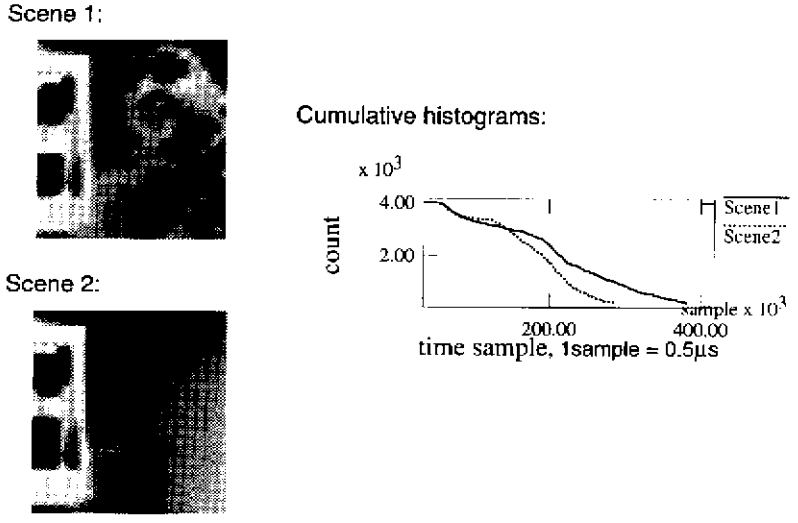


Figure 52: Detecting scene changes with the sorting computational sensor.

8.3.4 Image Segmentation

The cumulative histogram of an image can be used to segment an image into regions. Pixels from a single region often have pixels of similar intensity that appear as clusters in the image histogram [8]. The values which ought to be stored in the cells can be generated to correspond to the "label" of each such region. An off–chip global processor could perform this labeling by updating the supplied voltage when the transition between the clusters in the (cumulative) histogram is detected. An example of segmentation is shown in Figure 54b and Figure 54c in which the illuminated and shadowed regions respectively are labeled/colored as the black region.

8.3.5 Adaptive Dynamic Range Imaging

For faithful imaging of scenes with strong shadows, a huge dynamic range linear camera is needed. For example, the illumination of the scene which is directly exposed to the sunlight is several orders of magnitude greater than the illumination for the surfaces in the shadow. Due to the inherently large

Figure 53: A scene with back lit objects as captured by a conventional CCD camera. The box is roughly the field of view of the sorting sensor.

dynamic range of the sorting sensor, both illuminated and shadowed pixels can be mapped to the same output range during a single frame.

We demonstrate this concept with back illuminated objects. Figure 53 shows a global view of this scene as captured by conventional CCD camera. Due to the limited dynamic range of the CCD camera, the foreground is

poorly imaged and is mostly black. (The white box roughly marks the field–of–view for the sorting sensor.)

However, when the scene is imaged with the sorting sensor (Figure 54a), the

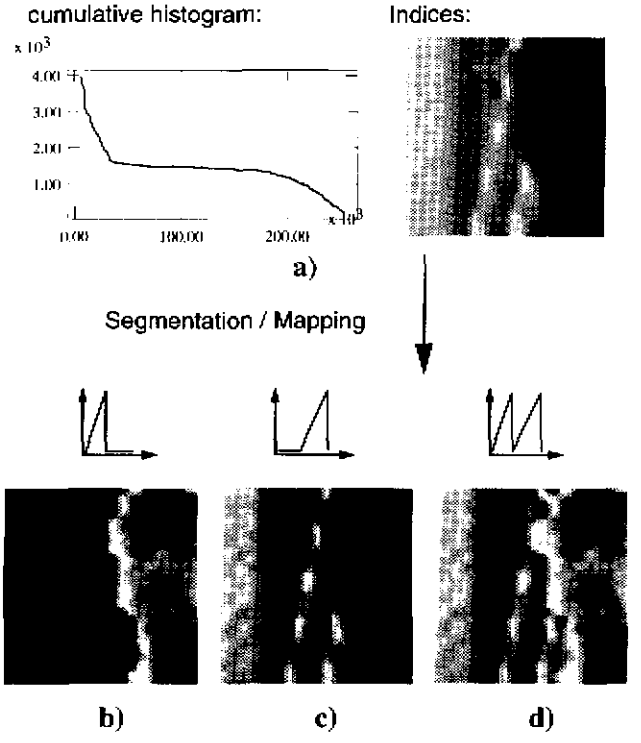


Figure 54: Sorting sensor processing: a) data from the sensors; b) segmentation (viewing the shadowed region); c) segmentation (viewing illuminated region); d) segmentation and shadow removal.

detail in the dark foreground is resolved, as well as the detail in the bright background. Since all 546 indices are competing to be displayed within 256 levels allowed for the postscript images in this paper, one enhancement for purpose of human viewing is to segment the image and amplify only dark pixels. The result is shown in Figure 54b. Conversely, as shown in Figure 54c, the bright pixels can be spanned to the full (8 bit) output range. Finally, if these two mappings are performed simultaneously the shadows are removed (Figure 54d.)

The same method can be obviously applied to the image obtained from a standard CCD camera. If the CCD image of Figure 53 is cropped to the

white box, and such an image is histogram equalized, we arrive at the result shown in Figure 55a. This image is analogous to the image of indices

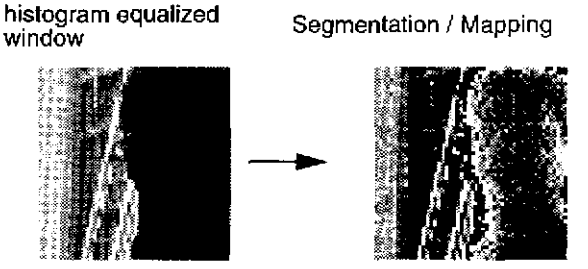


Figure 55: Conventional CCD camera processing: a) histogram equalization of the window; b) segmentation and shadow removal.

obtained by the sorting sensor (Figure 54a.) Due to the limited dynamic range, noise and quantization, the CCD image only resolves the face with 2–3 bits. The histogram equalized image from the CCD is used for further mapping using the same steps as for Figure 54d. The obtained result is shown in Figure 55b. Due to the obvious reasons, the result is poor. In contrast, the sorting computational sensor allocates as many indices, or output levels, as there are pixels within the dark region (or the entire image for that matter). By comparing Figure 54d and Figure 55b, the superior utilization of the sensory signal with the sorting chip is obvious.

The global quantity — a cumulative histogram waveform — is used internally to update local information within the cells. This is a top–down feedback that provides a more robust image representation as the light is being sensed. The cooperative massive parallelism together with the top–down feedback for adaptation clearly produced better results than conventional systems.

8.4 Relation to Computer Science

The process of sorting, or ordering, a list of objects according to some linear order such as $\leq$ for numbers is so fundamental and is done so frequently, that the subject has received considerable attention in computer science [2]. Traditionally, objects to be sorted are records consisting of one or more fields. One field in the record, called *the key*, is of a type for which a linear

ordering relationship $\leq$ is defined. Numbers and strings of characters are common examples of such types. The sorting problem is to arrange a sequence of records so that the values of their key fields form a non–decreasing (or non–increasing) sequence. The records all need not have distinct values, nor is it required that records with the same key value appear in any particular order.

The simplest algorithms usually take $O(n^2)$ time to sort n objects on a serial computer and are only useful for sorting a short list. One of the most popular sorting algorithm is quicksort, which takes $O(n \log n)$ time on average. It works well for most applications although its worst case performance is $O(n^2)$. There are other methods, such as heapsort and mergesort, that take $O(n \log n)$ time in the worst case, although their average behavior may not be quite as good as that of quicksort.

Under certain limitations there are sorting algorithms that sort in $O(n)$ time. Counting sort, for example, works for integers of known range. The counting sort first computes histogram of the input operands. The histogram is then used to compute a cumulative histogram. Finally, the cumulative histogram is used to determine for each input operand the number of inputs that have smaller (or greater) magnitude. This provides the index for each particular input. It is interesting to note the similarities between this algorithm and the way the sorting computational sensor sorts the input operands. However, a major difference is that the sorting sensor operates on analog or floating point values.

There are other sorting algorithms that have been developed for parallel computers [9]. Keys are equally distributed among processors, sorted and then merged. Some of them perform well for a small number of keys per processor, but most perform better if a number of keys per processor is large. This is not surprising, since the latter case performs most of the sorting internally; therefore, it requires less communication among the processors.

The sorting computational sensor that we have developed sorts keys which are supplied as an image. If this image is formed using some visualization device such as a TV monitor, it is possible to link record pointers to the corresponding (i,j) location in the formed image. Once sorting is performed the chip is read out in serial fashion. As an individual cell (i,j) is addressed and its order retrieved, the corresponding record can be sorted according to this index. Even though it may be argued that accuracy of this sorting processor is inferior compared to those implemented on a digital computer, sequences ordered in this way could be an excellent input for more rigorous

sorting on a digital computer. Sorting algorithms that perform best–on–average are those that use statistical properties of an initial record distribution. Using a preprocessing step employing the sorting computational sensor would enable such sorting algorithms to perform optimally every time.

As a crude comparison of performance, consider a 512 by 512 sorting computational sensor. Well illuminated input images could be sorted at 30 frames per second or more. If this load is supplied to the 32K Connection Machine CM–2 and if the best performing sorting algorithm — bitonic sort — is used, the sorting would be performed at 2 frames per second [9]. The inferior accuracy of the sorting computational sensor could very well be offset by its negligible cost and fast performance.

Chapter 9

Conclusion

The performance of existing machine vision systems still significantly lags that of a biological vision. The two most critical features presently missing from the machine vision are *low latency processing* and *top–down sensory adaptation*. The main contribution of this thesis is towards overcoming these two deficiencies by *implementing global operations in computational sensors*. Additional aims are to produce task–oriented self–contained machine vision components that can be used by a machine for a coherent interaction with the environment.

9.1 Global Operations and Computational Sensors

Computational sensors [37] incorporate computation at the level of sensing and have potential to both reduce latency and facilitate top–down sensory adaptation. First, by integrating sensing and processing on a VLSI chip both transfer and computational bottlenecks can be alleviated: on–chip routing provides high capacity transfer, while an on-chip processor may implement massively–parallel fine–grain computation providing high processing

capacity which readily scales up with the image size. Second, the tight coupling between processor and sensor provides opportunity for a fast processor–sensor feedback for top–down sensory adaptation.

In the context of this thesis the global operations are important for two reasons. First, in perception it seems that each important decision is a kind of global, or overall, conclusion about a perceived world. These conclusions are often what a machine needs for coping with a task at hand. The global operations thus can be considered to produce the ultimate goals of the vision processing for the coherent interaction with the environment. Second, global operations produce *a few* quantities for the description of the environment. Therefore, these quantities can be quickly transferred and/or processed to produce an appropriate action for a machine. In addition, the results of the global operations can be used within the computational sensor in top–down sensory adaptation thus directing a further processing for more reliable performance.

Global operations, however, need to gather and process information over the entire set of data. This global exchange of data among a large number of processors/sites quickly saturates communication connections and adversely affects computing efficiency in parallel systems — parallel digital computers and computational sensors alike. It is not surprising that there are only a few computational sensors which implement global operations, all with modest capability and/or low resolution [18] [66] [67]. On the other hand, there are many computational sensory which implement local operations [7] [30] [38] [47] [49] [68] [74], as those operations use only operands within a small spatial neighborhood of data and thus land themselves to the graceful implementation in VLSI. Local operations produce preprocessed images; therefore, a large quantity of data still must be read out and further inspected before a decision for an appropriate action is made — usually a time consuming process. Consequently, a great majority of computational sensors built thus far are limited in their ability to quickly respond to changes in the environment.

Implementing global operations in parallel systems has been the subject of extensive research in both computer engineering and computer science. The main difficulty with implementing global operations comes from the necessity to bring together, or aggregate, all or most of the data in the input data set. This work proposes two mechanisms for implementing global operations in computational sensors: (1) *sensory attention*, and (2) *intensity–to–time processing paradigm.*

9.2 Significance of the Sensory Attention

The sensory attention is based on the premise that salient features within the retinal image represent important global features of the entire data set. Then by selecting a small region of interest around the salient feature for subsequent processing, some global conclusions about the retinal image can be made. The sensory attention eliminates extraneous information and allows the processor to handle small amount of data at a time. This protects the limited communication and computation resources from information overload.

The sensory attention clearly mimics the process of visual attention in higher centers of the brain: a small interesting portion of the retinal image is selected to which the higher level processes can be restricted. Unlike eye movement, the attention shifts do not require any motor action, but occur internally on a fixed retinal image. For this reason, attention shifts are faster and play an important role in low–latency vision systems.

In our computational sensor implementation of the sensory attention, the saliency map is delivered optically by focusing an image onto the array of photodetectors. The photodetectors feed signals to the winner–take–all network which selects one salient location within the retinal image. The salient features which attract attention are bright spots in the retinal image. This particular embodiment of the sensory attention is called the *tracking computational sensor* — a VLSI sensor that attends onto and tracks a visual stimulus.

The tracking computational sensor operates in two modes: select mode and tracking mode. In the select mode the sensor detects the global intensity peak within a programmable active region, a subregion of the retina. This enables a user to define parts of the retinal image and aid the sensor to attend to the parts of image that are relevant for a task at hand. In the tracking mode the sensor dynamically defines its own active region, thus causing the sensor to ignore all retinal inputs except the currently tracked feature and its immediate neighborhood. This ensures that interference form the irrelevant information within the retinal image does not interfere with the currently attended information, i.e. the information important for the task at hand. In the tracking mode the sensor effectively remains locked on the selected feature and maintains the location of attention in the environmental coordinates rather then the image coordinates.

The significance of our implementation of the sensory attention is summarized as follows:

- The global data — the position and intensity of the feature — are easily and quickly routed from the chip via several output pins.
- In the tracking mode these global data are also used internally for the self–defined active region. This represents an example of sensor/processor feedback presently missing in artificial vision systems. The tracking computational sensor demonstrates the significance of this feedback, as it is essential in preventing erroneous information from interfering with the currently attended salient feature relevant to a task at hand.
- In the select mode, the sensor can restrict its operation to an arbitrary size region of interest. In combination with a clever image formation, this renders the sensor useful in a range of practical applications.
- Inherent in our implementation is the ability of the sensor to provide random access to the image data if needed. The image data can be read from a random location within the retinal image including the vicinity of the feature being tracked.
- The size of a cell in a conventional 2μ CMOS technology is 62μ by 62μ, which is about equivalent to the area taken by a 4x4 pixel region in an industrial CCD camera. This is an appreciable spatial resolution, especially given the versatility of functions performed by the sensor.

9.3 Significance of the Intensity–to–Time Processing

The other mechanism proposed for the implementation of global operations is the *intensity–to–time processing paradigm* — an efficient solution for massively parallel global computation over large groups of fine–grained data [12]. Inspired by the human vision, the intensity–to–time processing paradigm is based on the notion that stronger signals elicit responses before weaker ones. Assuming that the inputs have different intensities, the responses are ordered in time and a global processor makes decisions based

only on a few inputs at a time. The more time allowed, the more responses are received, thus the global processor incrementally builds a global decision based on several, and eventually on all of the inputs. The key is that some preliminary decisions about the retinal image can be made as soon as the first responses are received. Therefore, this paradigm has an important place in low–latency vision processing.

The intensity–to–time processing paradigm has been used to implement a *sorting computational sensor* — an analog VLSI sensor which is able to sort all pixels of an input image by their intensity while the image is being sensed. By the time all the inputs responded, the sensor has built an *image of indices*. The image of indices represents the histogram equalized version of the retinal image. In many computer vision applications the histogram equalization is the first image preprocessing operation performed on camera images, primarily for signal normalization and contrast enhancement.

During the computation, the global processor generates a waveform which is essentially the cumulative histogram of the retinal image. This waveform is one important global property of the retinal image which is reported with low latency on one of the output pins before image is ever read out.

The significance of the sorting computational sensor are summarized as follows:

- The global information — a cumulative histogram of the sensed scene — is reported on an output pin with low–latency.
- This global information is used internally within the computational sensor to generate the image of indices. This is an example of the top–down processor–sensor feedback.
- The image of indices has uniform histogram; therefore, (1) the dynamic range of the output circuitry is most optimally utilized from information theoretic point of view, and (2) the contrast is maximally enhanced.
- Histogram equalization is often the first processing step in image processing. The sorting sensor preforms this operation in the analog domain at the sensory level. Therefore, the sensory signal suffers less noise corruption caused by the signal transfer and quantization.
- The image of indices never saturates. This is a better scheme for preventing saturation than the logarithmic photo detection proposed by other researchers [10] [53].
- The cell size of the sensor in 2μ CMOS technology is 76μ by 90μ. A sensor in 1.2μ CMOS technology is currently being fabricated with the cell size of 38μ by 38μ, which is about the size of a 3 by 3 pixel region in an industrial CCD camera. This is an appreciable spatial resolution for a sensor which implements a global operation on a massive amount of input data.

9.4 Future

It is generally believed that the future of computing depends on the exploitation of large–scale parallel processing. Although specialized parallel computers have been successfully used in many different application areas, there remain significant obstacles to the widespread use of parallel computers in task–oriented machine vision. The most significant obstacles include the large size, power consumption and cost. The computational sensors proposed by this thesis are implemented in commodity VLSI tech-

nology. There is a strong indication that this technology will remain dominant technology for many years. Furthermore, the cost of the technology will continue to go down, while its capabilities will continue to improve.

Tree–dimensional multi–chip packaging and through wafer interconnects are gaining increasingly more interest in VLSI community and will probably be available within few years. Then, one may imagine most of the low–level machine vision processing being implemented within a three–dimensional stack of computational sensor chips (Figure 56). Many of these chips may implement various local operations as the information traverses through the stack of chips. Computational sensors performing global operations, however, will be essential at the higher levels of the stack. They will allow the results to be quickly routed off the stack for further high–level reasoning.

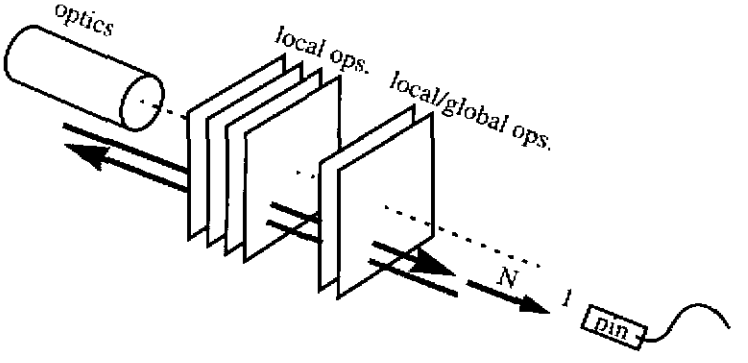


Figure 56: Computational Sensor Vision System.

When this concept becomes possible, the low–latency, robust performance, low power and portability will make many new applications for machine vision possible. Given the current trend, the area which will probably be the most dramatically impacted is a human–machine interaction. Humans will start seeing increasingly more vision based systems *around* and *on* themselves: in their homes, cars, offices, hospitals, entertainment, computers, etc. Therefore, the future of computational sensor seems promising. The low–latency computational sensors performing global operations on massive amount of data will find important place in that future.

Bibliography

[1] Abu–Mostafa, Y.S. "Lower Bound for Connectivity in Local–Learning Neural Networks," Jour. Complexity, Vol.4, pp. 246-255, 1988.

[2] Aho, A.V., J.E. Hopcroft, J.D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, 1983.

[3] Allport, A. "Visual Attention", Foundationa of Cognitive Science, M. Posner (ed.), MIT Press, 1989, pp. 631–682.

[4] Aloimonos, J. (edt.), *Special Issue on Purposive, Qualitative, Active Vision*, CVGIP: Image Understanding, Vol. 56, No. 1, 1992.

[5] A.G. Andreou, at al, "Current-Mode Subthreshold MOS Circuits for Analog VLSI Neural Systems," *IEEE Transactions on Neural Networks*, Vol. 2, No. 2, pp. 205-213, March 1992

[6] Anton, P. et al. "Temporal Information Processing in Synapses, Cells, and Circuits", Single Neuron Computation, Academic Press, 1992, pp. 291–313.

[7] W. Bair, and C. Koch. "An Analog VLSI Chip for Finding Edges from Zero-crossings." In: *Advances in Neural Information Processing Systems*, Lippman, R., Moody, J., and Touretzky, D.S., eds., Vol. 3, pp. 399-405, Morgan Kaufmann, San Mateo, CA, 1991.B.

[8] Ballard, D.H. and C.M. Brown, *Computer Vision, Prentice-Hall*, 1982.

[9] Blelloch, G.E. at al. "A Comparison of Sorting Algorithms for the Connection Machine CM–2", SPAA, 1991.

[10] Boahen, K. " Retinomorphic Vision Systems" *MicroNeuro 1996*, Lausanne, Switzerland. February 1996. To be published by IEEE Comp. Soc. Press.

[11] Brajovic, V. and T. Kanade "Computational Sensors for Global Operations," Proc. of ARPA IU Workshop, 1994.

[12] Brajovic, V. and T. Kanade "A Sorting Image Sensor: An Example of Massively Parallel Intensity-to-Time Processing for Low-Latency Computational Sensors," Sumbitted to the *Inter. Conf. Robotics and Automation*, Aprl 24-26, 1996, Minneapolis, MN, USA.

[13] Burgi, P.Y., T. Pun, "Asynchrony in image analysis: using the luminance–to–response–latency relationship to improve segmentation," *J. Opt. Soc. Am. A*, Vol.11, No. 6, June 1994, pp. 1720-1726

[14] Burt, P.J. "Smart Sensing within a Pyramid Vision Machine", IEEE Proceedings, Vol.76, No. 8, August 1988.

[15] C. P. Chong, A.T. Salama and K.C. Smith, "Image-Motion Detection Using Analog VLSI," *IEEE Jour. Solid-State Circuits*, Vol.27, No. 1, pp. 93-96, January 1992.

[16] Cormen, T.H. , C. Leiserson, R. Rivest, *Introduction to Algorithms*, MIT Press, 1990.

[17] DeWeerth, S.P., "Analog VLSI Circuits for Stimulus Localization and Centroid Computation," *Intl. Jour. of Comp. Vision*, Vol. 8, No. 3, 1992, pp. 191-202.

[18] DeWeerth, S.P. and C.A. Mead, "A Two–Dimensional Visual Tracking Array," *Proc. of the 5th MIT Conf. on Adv. Research in VLSI*, pp. 259–275, March 1988.

[19] Dowling, J.E.*The Retina: An Approachable Part of the Brain,*" Harward University Press, 1987.

[20] Van Essen, D.C, et al. "Pattern recongition, Attention, and information Bottlenecks in the primate visual system" in *Visual Information*

Processing: Form Neurons to Chips, SPIE, Vol. 1473, 1991 pp. 17-28.

[21] R.R. Etienne-Cummings, S.A. Fernando, J. Van der Spiegel, "Real-Time 2-D Analog Motion Detector VLSI Circuit," *Proceedings of the 1992 IJCNN Conf.*, Baltimore, MD, June 7-11, 1992.

[22] R.R. Etienne-Cummings, S.A. Fernando, J. Van der Spiegel, "A New Temporal Domain Optical Flow Measurement Technique for Focal Plane VLSI Implementation," *Computer Architecture for Machine Vision Conf.*, New Orleans, LO December 15-17 1993.

[23] Feldman, J.A., and Ballard, D.H., "Connectionist models and their properties," cog. Sci. Vol 6, pp. 205-254, 1982.

[24] Fossum, E.R., "Charge–coupled Computing for Focal Plane Image Preprocessing," Optical Engineering, pp. 916–922, Vol. 26. No.9,September 1987.

[25] A. Gruss, L.R. Carley and T. Kanade, "Integrated Sensor and Range-Finding Analog Signal Processor," *Jour. Solid-State Circuits*, Vol. 26, No. 3, pp. 184-191, March 1991.

[26] A. Gruss, S. Tada, and T. Kanade, "A VLSI Smart Sensor for Fast Range Imaging", Proc IEEE International Conference on Intelligent Robots and Systems (IROS'92), Raleigh NC, July 7-10, 1992.

[27] M. Hakkarainen and H.-S. Lee, "A 40x40 CDD/CMOS Absolute-Value-of-Difference PRocessor for Use in a Stereo Vision System" *Jour. Solid-State Circuits*, Vol. 28, No. 7, pp. 799-807, July 1993.

[28] M. Hakkarainen, J. Little, H.-S. Lee and J.L. Wyatt, Jr., "Interaction of Algorithm and Implementation for Analog VLSI Stereo Vision," *SPIE Int'l Symp. on Optical Eng. and Photonics in Aerospace Sensing*, Orlando, FL, pp. 173-184, April, 1991.

[29] Hamamatsu Photonics K.K, "Position Sensitive Detectors", Hamamatsu City, Japan.

[30] J.G. Harris, C. Koch, and J. Luo, "A two-dimensional analog VLSI circuit for detecting discontinuities", Science, Vol. 248, pp 1209-1211, 1990.

[31] J. Harris, S.C. Liu and B. Mathur, "Discarding Outliers Using a Non-linear Resistive Network," *Proc. IJCNN, Vol. I*, pp. 501-506, Seattle, Wash., July 1991.

[32] Holden, A.L. "The Central Visual Pathways" in *The Eye, Vol. 2A Visual Function in Man*, Academic Press , 1976

[33] T. Horiuchi, J. Lazzaro, A. Moore, C. Koch "A Delay-line Based Motion Detection Chip," in *Advances in Neural Information Processing Systems Vol. 3*, Lippman, R., Moody, J., Touretzky, D., eds., pp. 406-412, Morgan Kaufmann, San Mateo, CA 1991.

[34] Horn, B., *Robot Vision*, MIT Press, 1986.

[35] Horn, B., *"Parallel Networks for Machine Vision,"* A.I. Memo No. 1071, MIT, December 1988.

[36] Hwang, K. and F.A. Briggs, *Computer Architecture and Parallel Processing*, MCGraw–Hill, 1984

[37] Kanade, T. and R. Bajcsy, "Computational Sensor: Report form the DARPA Workshop", IUS Proceedings, pp. 335–350, 1993.

[38] C.L. Keast and C.G. Sodini, *"A CCD/CMOS–based Imager with Integrated Focal Plane Signal Processing,"* IEEE Journal of Solid–State Circuits, Vol.28, No. 4, April 1993.

[39] C.L. Keast, and C.G. Sodini, "A CCD/CMOS Process for Integrated Image Acquisition and Early Vision Signal Processing," *Proc. SPIE Charge-Coupled Devices and Solid State Sensors*, Santa Clara, CA, pp. 152-161, February 1990.

[40] C. Koch, *"Seeing Chips: Analog VLSI Circuits for Computer Vision"*, Neural Comp. 1: 184–200, 1989.

[41] Koch, C. "Implementing early vision algorithms in analog hardware: an overview" in *Visual Information Processing: Form Neurons to Chips*, SPIE, Vol. 1473, 1991 pp. 2–16.

[42] Koch, C. and S. Ullman, "Selecting one among the many: A simple network implemnting shifts in selective visual attention," *Human neurobiol.* Vol 4, 1985, pp. 219-227.

[43] Koch, C. and S. Ullman, "Shifts in Selective Visual Attention: Toward the Underlying Neural Circuitry. In L.M. Vaina (edt.), *Matters of Intelligence*, Reidel Publishing, 1987, pp. 115-141.

[44] G. Kreider, J. Van der Spiegel, I. Born, C. Claeys, I. Debusschere, S. Sandini, P. Dario and F. Fantini, *"A Retina–Like Space Variant CCD Sensor,"* Proc. SPIE Conf. on CCD and Solid State Optical Sensors, VOl. 1242, pp.133-140, Santa Clara, CA, February 12-13, 1990.

[45] Kuo, A., "A VLSI System for Light–Stripe Range Imaging", M.S. Thesis, ECE Dept., Carnegie Mellon University, Pittsburgh, PA, December 1992.

[46] J. Lazzaro, S. Ryckebusch, M.A. Mahowald and C. Mead, "Winner-Take-All Networks of O(n) Complexity," in *Advances in Neural Information Processing Systems Vol. 1*, D. Tourestzky, ed., pp. 703-711, Morgan Kaufmann, San Mateo, CA, 1988.

[47] Maher, M.C., S. DeWeerth, M. Mahowald, C. Mead "Implementing neural Architectures Using Analog VLSI Circuits", *IEEE Trans. Circuits and Systems*, Vol. 36, No. 5, May 1989.

[48] M.A. Mahowald and T. Delbruck, "Cooperative Stereo Matching Using Static and Dynamic Image Features," in *Analog VLSI Implementation of Neural Systems*, C. Mead and M. Ismail, eds., Kluwer, Boston, pp. 213-238, 1989.

[49] M.A. Mahowald and C. Mead, "Silicon Retina," Chap. 15 of *Analog VLSI and Neural Systems*, Addison-Wesley Publishing Co., 1989.

[50] Mahowald, M.A., "Evolving Analog VLSI Neurons," in Single Neuron Computation, Ed. McKenna, Davis and Zornetzer, Academic Press, 1992.

[51] MasPar Computer Corporation, 749 North Mary Avenue, Sannyvale, California 94086, U.S.A.

[52] B. Mathur and C. Koch, *Visual Information Processing: From Neurons to Chips*, eds., *Proc. SPIE*, Vol. 1473, 1991.

[53] C. Mead, *Analog VLSI and Neural Systems*. Reading, MA: Addison-Wesley, 1989.

[54] R. Milanese, *"Detecting Salient regions in an Image: from Biological evidence to computer implementation"*, Ph.D. Thesis, Dept. of Computer Science, U. of Genova, Switzerland, December 1993.

[55] Mortara, A. et al. "A communication Scheme for Analog VLSI Perceptive Systems", IEEE Jour. of Solid–State Circ. Vol.30, No. 6, June 1995, pp. 660–669

[56] Muroga, S. and Toda, I. "Lower Bound of the Number of Threshold Functions," IEEE Trans. Elec. Comp, EC–15, pp. 805–806, 1966.

[57] Neisser, U.: *Cognitive Psychology*, Appleton, New York, 1967

[58] T. Nishimura, et. al. "Three Dimensional IC for High Performance Image signal Processor," *Proc. of the IEDM Int. Elec. Dev. Meeting*, Washington, D.C. December 6-9, 1987.

[59] Pan T–W. and A.A. Abidi, "A 50dB Variable Gain Amplifier Using Parasitic Bipolar Transistors in CMOS," IEEE Jour. of Solid–State Circuits, pp. 951–961, Vol. 24, No. 4, August 1989.

[60] Pulfrich, C. "Die Stereoskopie im Dienste der isochromen und heterochromen Photometrie," *Naturwissenschaften*, Vol. 10, pp. 553-564, 1922.

[61] Ripps, H. and R.A. Weale, "Temporal Analysis and Resolution" in *The Eye*, Vol. 2A, ed. H. Davson, Academic Press, 1976, pp. 185-217.

[62] Rosenblatt, F. *The Perceptron: The probablilistic model for information storage and organization in the brain*, 1958

[63] Shannon, C. E. and W. Weaver, *The Mathematical Theory of Communications*, Univ. Illinois Press, Urbana, IL, 1949

[64] J. Van der Spiegel, G. Kreider, C. Claeys, I. Debusschere, G. Sandini, P. Dario, F. Fantini, P. Bellutti and G. Sondini, "A Foveated Retina-Like Sensor Using CCD Technology, "Chap. 8 of *Analog VLSI Implementation of Neural Systems*, C. Mead and M. Ismail, eds., Kluwer Academic Publishers, 1989.

[65] Sivilotti, M.A., M.R. Emerling and C.A. Mead, "VLSI Architectures for Implementation of Neural Networks", Conf. on Neural Networks for Computing, pp. 408–413, Snowbird, UT, 1986.

[66] D. Standley, "An Object Position and Orientation IC with Embedded Imager," *IEEE Journal of Solid-State Circuits*, Vol. 26, No. 12, pp. 1853-1860, 1991.

[67] J. Tanner and C. Mead, "Optical motion sensor." in *Analog VLSI and Neural Systems*, C. Mead, pp. 229-255, Addison-Wesley, Reading, MA, 1989.

[68] Tremblay M., Laurendeau D. and Poussart D. "High resolution smart image senosr with integrated parallel analog processing for multiresolution edge extraction", Robotics and Autonomous Systems, Vol. 11, 1993, pp. 231–242.

[69] G.W. Turner, C.K. Chen, B.-Y. Tsaur, and A.M. Waxman, "Through-Wafer Optical Communication Using Monolithic InGaAs-on-Si Led's and Monolithic PtSi-Si Schottky-Barrier Detectors," *IEEE Photonics Technology Letters*, Vol. 3, No. 8, August 1991.

[70] W. B. Veldkamp and T. J. McHugh, "Binary Optics," *Scientific American*, May 1992.

[71] Weckler, G.P. "Operation of p-n Junction Photodetectors in a Photon Flux Integrating Mode," IEEE Jour. of Solid–State Circuits, pp. 65–73, Vol. sc–2, No. 3, September, 1967

[72] J.L. Wyatt Jr., D. Standley and B. Horn, "Local Computation of Useful Global Quantities Using Linear Resistive-Grid Networks," poster session, *Conf. on Neural Networks for Computing*, Snowbird, UT, April 1990.

[73] C. Yarlagadda, "A 512x512 Random Addressable Variable Resolution Image Sensor," *M.S. Thesis*, Dept. of Elec. and Comp. Eng. New Jersey Institute of Technology, 1990.

[74] P.C. Yu, S.J. Decker, H.S. Lee, C.G. Sodini and J.L. Wyatt, Jr., "CMOS Resistive Fuses for Image Smoothing and Segmentation," *IEEE Journal of Solid-State Circuits*, Vol. 27, No. 4, pp. 545-553, April 1992.