AFRL-RI-RS-TR-2013-125

FOUNDATIONS OF NEUROMORPHIC COMPUTING

MAY 2013

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW, Wright-Patterson AFB Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2013-125 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/ S /

JOSEPH A. CAROLI
Chief, High Performance Systems Branch

/ S /

MARK LINDERMAN
Technical Advisor ,Computing & Communications Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
MAY 2013	FINAL TECHNICAL REPORT	SEP 2009 – SEP 2012

4. TITLE AND SUBTITLE

FOUNDATIONS OF NEUROMORPHIC COMPUTING

5a. CONTRACT NUMBER
IN-HOUSE

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

AFRL/RITB:
Clare Thiem, Bryant Wysocki, Morgan Bishop, Nathan McDonald

Rome Research Corp:
James Bohl

5d. PROJECT NUMBER
NEUR

5e. TASK NUMBER
PR

5f. WORK UNIT NUMBER
0J

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RITB
525 Brooks Road
Rome, NY 13441-4505

Rome Research Corporation
421 Ridge St.
Rome, NY 13440

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RITB
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2013-125

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. PA# 88ABW-2013-2241
Date Cleared: 9 May 2013

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This neuromorphic computing research and development effort explored the design and implementation of computationally intelligent computer architectures and high performance computer software algorithms. The in-house research concentrated on the design of mathematical models, algorithms, computing architectures, and computational efficiencies for the advancement of neuromorphic computing and neuroprocessors. The software aspect of the research demonstrated the combination of computational power with human level cognitive functionality to create a system that can increase Department of Defense (DoD) operators' and analysts' ability to analyze textual and character data. The hardware aspect of the research explored memristor-based and zero instruction set computing technology to provide neuromorphic computing for size, weight, and power limited applications. Progress was made on both the hardware and software side of neuromorphic computing research setting the stage for future agile information systems.

15. SUBJECT TERMS
Neuromorphic Computing, Confabulation, Memristor, Reasoning, Perception, Artificial Neural Networks, Modeling, Simulation, Optic Flow, Zero Instruction Set Computing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**JOSEPH A. CAROLI**
U	U	U	SAR	48	19b. TELEPONE NUMBER (*Include area code)* **315-330-4893**

TABLE OF CONTENTS

Section Page

LIST OF FIGURES

LIST OF TABLES

Tables Page

1.0 SUMMARY

This neuromorphic computing research and development effort explored the design and implementation of computationally intelligent computer architectures and high performance computer software algorithms. Neuromorphic computing is instrumental in the development of intelligent systems able to imitate natural neuro-biological processes. This is achieved by artificially recreating the highly parallelized computing architecture of the mammalian brain. In particular, neuromorphic systems have demonstrated potential for applications in pattern recognition and optimization such as behavior modeling, target finding, intelligent automated data processing, and intelligence analysis. This neuromorphic computing research exploited emerging computing architectures, the characteristic behaviors of novel complex materials, and the parallel nature of biological intelligence. The research concentrated on the design of mathematical models, algorithms, computing architectures, and computational efficiencies for the advancement of neuromorphic computing and neuroprocessors where the development of such systems draws from many fields. This multidisciplinary work was based on recent advancements in nanotechnology (Nano – e.g. nano-scale engineering and material science), high performance computing (HPC – e.g. computer engineering), electronics (e.g. design and fabrication of hybrid electronic systems) and neuroscience (NeuroSci – e.g. computational intelligence and cognitive psychology) as depicted in Figure 1. This research area focuses on developing enhanced autonomy and perception for use in Air Force systems.

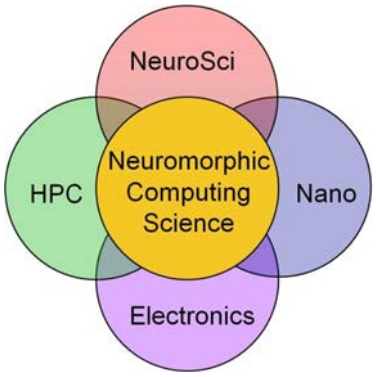


Figure 1. The multidisciplinary aspect of neuromorphic computing science

More specifically, the overall focus of the Information Directorate's in-house neuromorphic research was in application modeling and memristive system development for computing architectures that exploit the physical properties of nano-enabled materials and electronics for energy efficient computing and computationally intelligent systems. This work leveraged the previous in-house research on genetic algorithms and the concurrent nanotechnology effort which explored development and deployment opportunities in the emerging disciplines of nanoscience, nanoengineering, and nanobioscience, for advanced computing architectures.

The effort began with the examination of leading computational intelligence theories for implementation on massively parallel HPCs. Brain state in a box (BSB), Confabulation, and Spiky hierarchical temporal models were explored. A hybrid BSB/confabulation model was

developed and scaled to infer meaning from text documents with errors and occluded letters. The application was successfully expanded to include both English and Chinese characters.

The program also explored the physical nature of memristive systems in an effort to uncover the basic principles behind their operation and to aid in the design of improved synaptic devices. Both physical and compact models were developed as tools for memristive device design and for circuit simulation and testing. The models were compared to fabricated memristive systems and refined. In addition, the models were used to direct memristor design in an iterative process. Neuromorphic circuits were simulated, built, and tested demonstrating successful training and response of electronic neural circuits with synaptic weighting.

The program further examined the utility of application-specific integrated circuit (ASIC) - based artificial neural networks (ANNs) which are ideally suited for mobile or portable platforms with strictly limited size, weight, and power (SWaP) resources. Effort was expended to gain a better understanding of issues related to operating in a SWaP constrained environment. A fully parallel, silicon-based artificial neural network (CogniMem CM1K) built on zero instruction set computer technology was then used for change detection and object identification in video data and network intrusion detection for surveillance and cyber security applications.

In the end progress was made on both the hard and software side of neuromorphic computing research. The stage has been set for creating future agile information systems with more capability for reasoning and perception that will allow the command chain to make informed decisions quicker than our adversaries.

2.0 INTRODUCTION

The increasing resolution and speed of today's advanced sensor platforms provide an overwhelming and exponentially growing supply of data, which has subsequently created a demand for autonomous pattern recognition systems that scour raw or preprocessed data in an effort to extract meaningful information [1 - 3]. Such large amounts of data choke communications channels and restrict or prohibit real-time analysis. High performance computers and increased bandwidth can alleviate some of the strain but are only part of a more sophisticated solution. Computationally intelligent systems that convert data into information at the source of detection are required to limit the burden on communications, processing, and analytical resources so that relevant real-time analysis can occur. Massively parallel hardware based neuromorphic computing can achieve the size, weight, and power requirements needed for such platform applications while providing an intelligent scan of the data prior to transition. Rather than relying on brute force computations that require large amounts of resources and can be relatively slow due to their serial nature, neuromorphic systems provide a more elegant method for signature recognition and autonomous operations.

The improved processing power of modern high performance computers enables implementation of large, sophisticated pattern recognition systems based on statistical analysis and neural network schemes. But programmable machines are limited in their ability to address fuzzy combinatorially complex scenarios. Neuromorphic processors, which are based on the highly

parallelized computing architecture of the mammalian brain, show great promise in providing the environmental perception and comprehension required for true adaptability and autonomy and is thought to be better able to solve fuzzy perception and classification problems historically difficult for traditional, von Neumann-based computers.

Therefore, the research objective of this effort was to investigate, develop and demonstrate application of a scalable neuromorphic computer to Air Force (AF) relevant problems and applications such as intelligent autonomic, recognizant, and image processing systems. The goal of our research effort is to identify, evaluate, develop, and demonstrate prototype neuromorphic computer elements and circuit forms, working toward scalable systems that employ leading edge and emerging computing architectures (i.e. complementary metal-oxide semiconductor (CMOS), optical, thin film, or nanotechnology, i.e. memristor-based technologies) that will target the integration of thousands (x,000's) of neurons and millions of synapses (house fly, x=338 [4], honeybee, x=960 [5]). The thought is that this level of integration is required in order to carry out AF intelligence functions such as autonomous pattern, image, voice, video, and/or sound recognition.

3.0 METHODS, ASSUMPTIONS AND PROCEDURES

Work under this effort can be broken down into two broad areas of research, one of which was software focused and the other which was hardware focused. The software focused research pursued the development of algorithms and the utilization of high performance computing (HPC) assets for neuromorphic computing. The hardware focused research examined the development of hardware technology for neuromorphic computing in size, weight and power (SWaP) constrained environments which involved memristive device based technology, open source hardware, and off of the shelf technology such as zero instruction set computing (ZISC) technology [6].

3.1 Software Focused Research

The software focused area of this effort focused on the research and development of highly-scalable neuroscience inspired computing models, algorithms and architectures suitable for massively parallel computing systems for robust information extraction and exploitation. These models are currently being applied to occluded text recognition, autonomous motion detection, object recognition and predictive sensing.

The concept for intelligent textual extraction, recognition, and understanding on HPC systems is divided into 3 layers. This layered structure has some analogies to human information processing. Research discoveries in human psychology show that information is first processed by the sensory cortex where the complex data is reduced to abstract representations. The abstract representation is compared to stored patterns at the basal ganglia and neocortex to generate perception and quick reaction. If more sophisticated processing such as reasoning is needed then a relatively slow sequential process will occur in the prefrontal cortex. For the research activity covered by this section of the report, the character recognition layer serves as the interface with the physical world, the word and sentence confabulation layer generates knowledge based

perception of the abstract information coming from the lower layer. The output from the perception level is then fed into an information fusion framework in the top layer. The input of the system is the text image.

The first layer is the character recognition phase based upon Brain-State-in-a-Box (BSB) models. The BSB tries to recall the input image with stored image patterns of the English alphabet, punctuation and other entities in natural text. If there is noise in the image, multiple matching patterns may be found. For example, a horizontal scratch will make the letter "T" look like the letter "F". In this case this is ambiguous information. The ambiguity can be removed by considering the word level and sentence level context, which is achieved in the second and third layer where word and sentence recognition is performed using cogent confabulation models. The models fill in the missing characters in a word and missing words in a sentence. The three layers work cooperatively. The BSB layer feeds forward the character recall information to the word recognition layer while the word recognition layer feeds back word level context information that directs the BSB models to perform pattern matching more efficiently. The word recognition layer sends words and partial words to the sentence recognition layer while the sentence recognition layer also sends back sentence level context information that helps the word recognition layer to fill in missing characters more efficiently. Larger dimensional BSB models provide a means for enhancing character recognition. Finally, the third layer performs natural language processing (NLP) on the sentence level confabulated text. It implements part of speech (POS) tagging, full or chunk parsing, and relevant noun phrase to do entity extraction. In addition, the text and entity extraction algorithms work in parallel and a globally networked knowledge database uses a multi-dimensional index within a distributed relational framework to allow extra degrees of freedom and enable real-time data processing. This additional processing and storage of the text enables natural language queries to be formulated by a user against the data store to return concise, relevant, and consolidated information based on their information requests.

The hardware platform of the system is based on the 500 Tera Floating-point Operations Per Second (TFLOPS) Condor cluster. The cluster consists of 80 sub-clusters and each sub-cluster is composed of two Intel Xeon Hexa-core processors as the head node, 22 Sony PlayStation3 (PS3) computers based on IBM Cell-BE processor, and 2 NVIDIA GPGPUs. Each cell processor has one PowerPC processor and 6 synergistic processing elements (SPE). Each SPE processor is a self-contained vector processor that runs 4 floating point operations at 3.2 GHz. With 6 of these SPEs, a cell processor provides 192 GFLOPS performance. Each PS3 only has 256MB memory, which is relatively small comparing to the conventional desktop PCs. Overall, the 1,760 cell processors deliver 338 TFLOPS computing power and form the first layer hardware of the system. The Intel Xeon processor based headnodes naturally form the second layer. Each headnode has 12 cores and 24GB SMART memory. The memory access speed could reach as high as 2GB/s per core. The GPGPUs will be utilized as larger dimensional BSB models and placed in the feedback loop of the word and sentence level confabulation to further enhance character recognition. 78 of the GPGPUs are the NVIDIA Tesla C1060 model which has 240 cores running at 1.3GHz providing 933 GFLOPs. The other half are the NVIDIA Tesla C2050 model which has 448 cores running at 1.15 GHz providing 1.03 TFLOPs.

3.2 Hardware Focused Research

Neuromorphic computing seeks to mimic brain functions and efficiencies through parallel operation, reconfigurability, high density, and low power consumption. The natural ability of the brain to perform high numbers of complex functions in parallel is currently unmatched by even the fastest most powerful super computers. Neuromorphic computers promise to provide machines the ability to perform complex functions by mimicking the brain's engineering but building such systems represents a great technological engineering challenge. Software based implementations of neuromorphic computing have demonstrated limited brain functionality [7]. However, software based techniques require high performance computers for processing that in turn make their use within mobile and/or low cost systems impractical. In short, the development of hardware based neuromorphic computers will enable new solutions to traditionally difficult computer problems that involve high mathematical variance or fuzziness, such as pattern recognition, nonlinear modeling, and process prediction.

3.2.1 Memristive Device Based Technology

The amazing computing power of the brain is in part due to its highly parallelized interconnectivity amongst neurons through synaptic connections. The synaptic connection plays an important role in brain activity as these connections can be strengthened or weakened as the brain learns and stores knowledge within the system [7]. Neuron behavior has been characterized as an adding system that provides an output based on the sum of all inputs (through synaptic connections), and its connectivity to other neurons [8]. In addition, the amount of information or knowledge a neuromorphic computer can retain depends on the number neurons and synaptic connections within the system. For example, the brain of an ant is said to contain approximately 338,000 neurons [4]. Given the large number of neurons and synaptic connections (approximately 1,000 synaptic connections per neuron) required to design systems capable of mimicking practical brain functionality, it is important for the individual electronic devices to be small. This facilitates fabrication within a reasonable physical area, similar to how computer microprocessors are fabricated today. In this section, a physical description is presented by which three transistor synaptic circuit functions mimic brain synaptic behavior. The implementation of synaptic circuitry within an adding node and single transistor adding neuron (a simple neuromorphic computing architecture) is also demonstrated. As an example to show the utility of a reconfigurable logic gate, we demonstrate our neuromorphic architecture as a reconfigurable XOR logic gate.

Computer simulations based on the physical principles described above demonstrate the feasibility of the neuromorphic architectures [9]. Variations of neuromorphic computing architectures can be constructed employing transistors, memristors, and inverter circuits coupled to floating node neurons or thresholding neurons to perform the desired computations.

We define our synaptic system as a building block employing a variable resistor (either a CMOS transistor operating within the weak inversion and Ohmic region modes or a memristor) and inverter circuit elements [9]. Figure 2 depicts our complete synaptic system where (a) illustrates the circuit implementation of the synapse, and (b) illustrates the simplified circuit element form that will be employed when designing the neuromorphic network. From the figure, we can

observe that the output of the synaptic system is a function of the Vm potential that will either strengthen, weaken, or completely cut-off the connection between the synapse input (SI) and the CMOS inverting circuit operating within its transition bias point range.

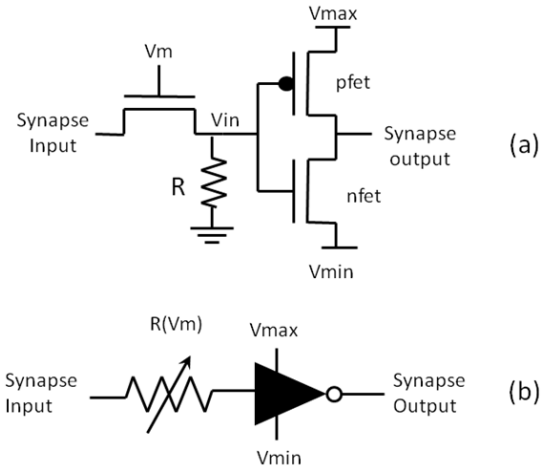


Figure 2. Synaptic system (a) circuit representation and (b) simplified circuit representation

We can model the synapse output (SO) employing a linear approximation to the electronic characteristic behavior of the CMOS inverting circuit as:

$$\mathrm{SO}(SI, Vm) = f[I(Vm) * R], \tag{1}$$

where f represents the transfer function of the CMOS inverter and I(Vm) is current across the transistor channel or memristor device, and R is a resistor used to reset and properly bias the synapse. For example, for synaptic inputs between 0 and 2 V, as shown in Figure 2, the inverter transfer functions can be made to range from 3 to ~0 V approximately as a function of the biasing operating voltage Vin as shown in Figures 2 and 3.

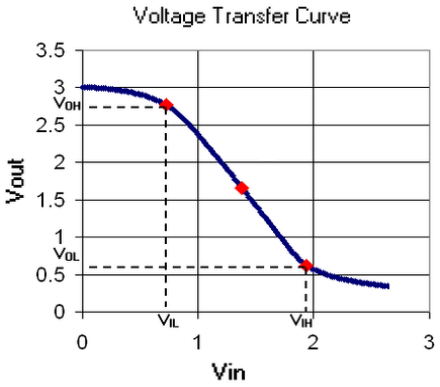


Figure 3. CMOS inverter voltage transfer curve.

The implementation of the neuron functionality will be performed with an adding node and/or a single transistor as displayed in Figures 4 and 5. The adding node is the physical connection where all post-synaptic outputs will converge. The combined response is fed to the next neuron synaptic layer of the neuromorphic computing architecture. As the synaptic outputs converge, they will increase or decrease the potential at the floating neuron adding node. Thus, the resulting added potential will become the input to the following synaptic layer. In addition, if the neuron adding node were connected to the gate of, for example, an NFET transistor, and if the total combined potential at the adding node is greater than the threshold voltage, Vth, of the MOSFET transistor, the output of the neuron will be Vo=Vn. The neuron output Vo will be fed to the next synaptic layer of the neuromorphic computing architecture. Otherwise, the neuron won't output a high potential (the neuron won't fire). Finally, Figure 6 displays the computing architecture required to implement the XOR function with our in-house developed neuromorphic computing architecture [9] classically described in [8]. Figure 6 is also an example of a neuromorphic computing concept that demonstrates how a computing architecture can be implemented with thresholding neurons.

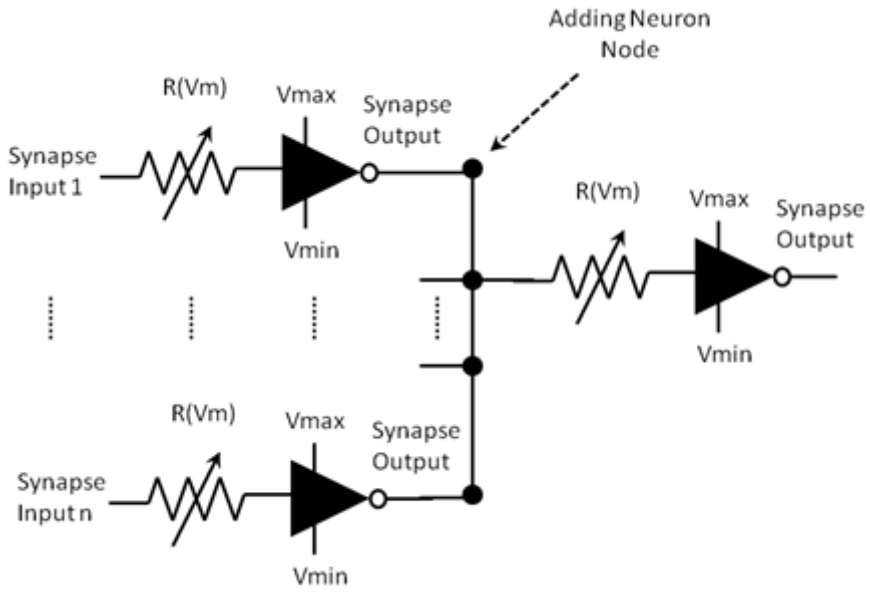


Figure 4. Multiple synaptic outputs converge at the floating adding node

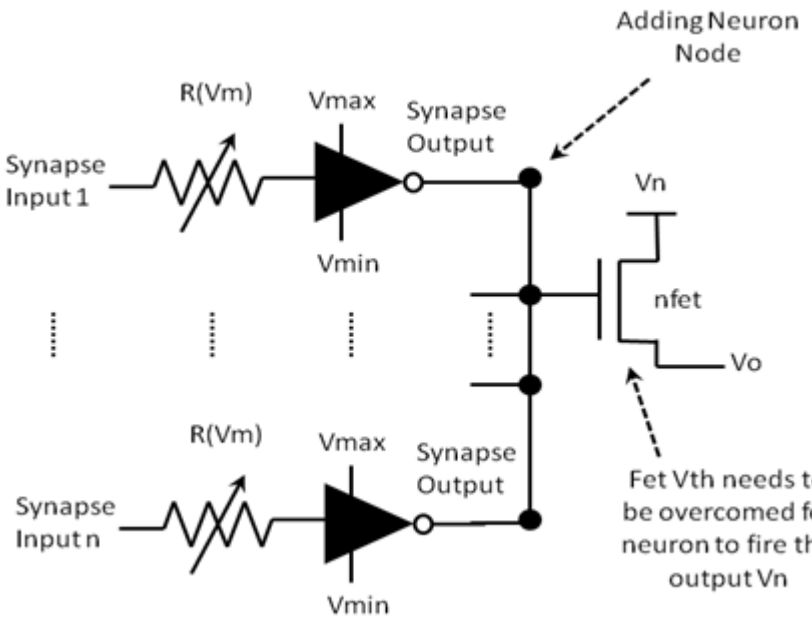


Figure 5. Multiple synaptic outputs converge at the adding neuron

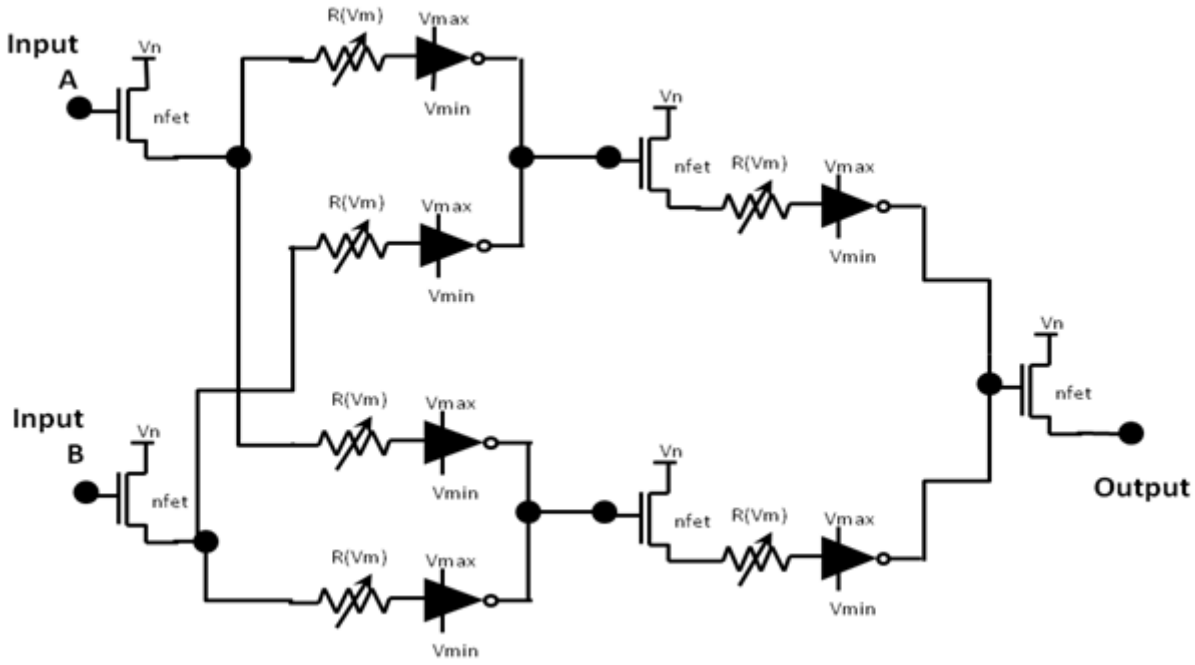


Figure 6. Neuromorphic computing implementation of the XOR function

Memristor Compact Modeling and Simulation

In order to successfully implement a memristor powered neuromorphic computer processor, it is important to be able to describe mathematically the electronic characteristic behavior of such devices to enable development and simulation of the computing architecture. Therefore, Air Force Research Laboratory Information Directorate's High Performance Systems Branch (AFRL/RITB) obtained access to memristor device hardware physical experimental data from Professor Kris Campbell's memristor research group at Boise State University. Utilizing this memristor hardware data, in-house memristor device compact models and simulation methodologies for chalcogenide memristor devices were developed. From a microprocessor design viewpoint, it is important to be able to simulate large numbers of devices within the integrated circuit architecture in order to reliably speed up the development process. Ideally, device models would accurately describe the characteristic device behavior and would be represented by single-valued equations without requiring the need for recursive or numerically intensive solutions. With this in mind, an empirical chalcogenide compact memristor model was developed that accurately describes all regions of operations of memristor devices employing single-valued equations [10].

The memristor device postulated in 1971 by Leon Chua [11] as the fourth basic circuit element has received much attention in the research community since the publication of Strukov's 2008 paper titled "The missing memristor found" [12]. The memristor name is a contraction for memory resistor because that is exactly its function: to remember its history [13]. The memristor is a two terminal passive device whose resistance state depends on its previous state and present electrical biasing conditions, and when combined with transistors in a hybrid chip, memristors could radically improve the performance of digital circuits without the need for further reduction of transistor dimensions [13]. Given their two terminal structural simplicity and electronic passivity, the applications for memristor technology range from non-volatile memory, instant on computers, reconfigurable electronics and neuromorphic computing [13,14]. According to Chua [14], the memristor behaves like a linear resistor with memory but also exhibits many interesting nonlinear characteristics, and several electronic models have been presented to describe the electrical behavior of memristor devices [11,12,14,15,16]. However, given that memristor devices are not commercially available, good physical model-to-hardware correlations have not yet been reported in the published literature. Therefore, in this work, we present what we believe to be the first model-to-hardware correlation of chalcogenide memristor electrical characteristics. In our studies, we have employed both linear and nonlinear models from the published literature to fit our memristor hardware. However, we have observed that these published models do not represent accurately the electrical characteristic behavior of our memristor device hardware. Therefore, we have developed a simple compact model that accurately represents the electrical behavior of chalcogenide based memristors. Our model consists of a threshold memristor model similar to those utilized in state-of-the-art CMOS modeling and simulation as, for example, the well-known BSIM4.6.4 from the University of California, Berkeley. These types of compact models have proven very valuable within the microelectronics industry given their straightforward single-valued mathematical formulations that make modeling and simulation within very large scale integrated circuits fast, reliable, and accurate. Common CMOS transistors operate within various regions of operation: Cutoff, Ohmic, and Saturation. Similarly, we have developed a memristor model that operates within three regions: off, nonlinear and on. It is our

goal to develop a compact memristor model that describes the electrical behavior of memristor devices, and it is simple and accurate to enable large scale device simulations.

All experimental hardware memristor devices studied were fabricated and tested at Boise State University, and the fabrication details have been described elsewhere [17]. The physical structure of the memristor devices characterized was comprised of three thin films of Ag, SnSe and Ge2Se3 sandwiched between tungsten metal contacts. All electrical DC experimental measurements were performed with an Agilent B1500A (HP4145B) semiconductor parameter analyzer and Micromanipulator 6200 microprobe station equipped with temperature controllable wafer chuck and tungsten probe W tips (Micromanipulator size 7A) at room temperature. The tested memristor devices were 180 nm in diameter with 80 μm x 80 μm tungsten pads for electrical contact to the top and bottom electrodes. Figure 7 displays the typical DC electrical Lissajous I-V curve characteristic response behavior of a memristor device under a sinusoidal input of 0.5 V amplitude and 100 Hz frequency. The memristor physical hardware experimental results were provided by Professor Kris Campbell, Boise State University. From the figure one can clearly observe that the memristor device toggles between two states of high and low conductivity. As the memristor device transitions from a low to high conductivity state it goes through high nonlinear diode-like processes at approximately -0.35 and 0.2 V threshold voltages respectively that switched the device from a low conductivity state to a high conductivity state and vice-versa. The threshold voltage analogy is used here to describe the voltage biasing region where nonlinear behavior occurs.

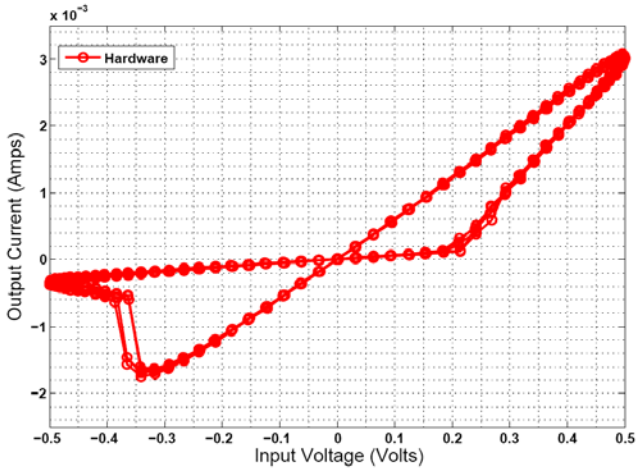


Figure 7. Typical memristor device DC electrical Lissajous I-V curve characteristic behavior

The most basic mathematical definition of a memristor is that of a current-controlled device for circuit analysis in the generalized class of nonlinear dynamical systems called memristive systems described by the equations [12,14]:

$$v = R(w, i)i \text{ , and} \tag{2}$$

$$\frac{dw}{dt} = f(w, i), \tag{3}$$

where w can be a set of state variables and R and f can in general be explicit functions of time [12,14]. For simplicity and ease of simulation, the memristor's resistance or memristance definition has been reduced to that of a current-controlled, time-invariant, one-port device given by [12]:

$$M(w) = \frac{w}{D} R_{on} + \left(1 - \frac{w}{D}\right) R_{off}, \tag{4}$$

where w represents the doped region of the memristor, D the total length of the memristor device, R_{on} the lowest resistance state and R_{off} the highest resistance state as graphically described in Figure 8.

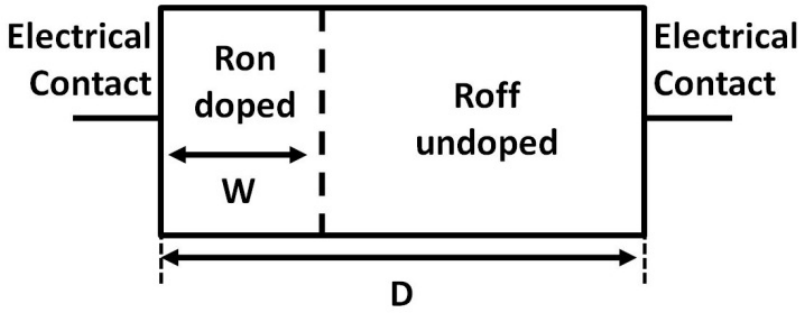


Figure 8. Memristor device physical structural diagram

In order to describe the velocity at which w increases, meaning the memristor device is becoming less resistive, Equation (3) can be described as [12]:

$$\frac{dw(t)}{dt} = u_v \frac{R_{on}}{D} i(t), \tag{5}$$

where u_v is the average ion mobility for the simplest case of ohmic electronic conduction and linear ionic drift in a uniform field [12].

Utilizing Ohm's law that states that the voltage across a resistor is directly proportional to the resistance times the current through the conductor, we can obtain from Equations (4) and (5) the following relationship [15]:

$$w(t) = u_v \frac{R_{on}}{D} q(t). \tag{6}$$

By inserting Equation (6) into Equation (4), we can obtain the memristance of the system, which for a R_{on} much less than R_{off} can be simplified to [12]:

$$M(q) = R_{off} \left(1 - u_v \frac{R_{on}}{D^2} q(t)\right). \tag{7}$$

Equation (6) describes the memristance of the memristor system as a function of the charge q(t). Additional improvements have been proposed to the aforementioned memristor model to include non-linear boundary conditions [15, 16]. The non-linear boundary condition proposed is of the form:

$$f_p(w) = 1 - \left(2\frac{w}{D} - 1\right)^{2p}. \tag{8}$$

The nonlinear window function in Equation (8) guarantees zero velocity of the doped/undoped barrier interface described in Figure 8 as *w* approaches either boundary, *w*=0 or *w*=D. Moreover, the differences between the models with linear and nonlinear drift disappear when *p* increases [11]. The incorporation of the window function described by Equation (8) requires the redefining of Equation (5) as follows [19]:

$$\frac{dw(t)}{dt} = u_v \frac{R_{on}}{D} i(t) f_p(w). \tag{9}$$

For p = 1, it is possible to integrate Equation (9) analytically; however, for values of *p* larger than 1 only numerical solutions are possible [19].

Attempts have been made to perform a model fit utilizing both the linear [12] and nonlinear [1, 15] memristor models. Figure 9 shows the memristor model-to-hardware correlation fit between experimental results and the linear memristor model, green line, described by [12] and nonlinear memristor model, magenta line, described by [16]. From the figure, we can observe that the linear model does not capture accurately the high memristor nonlinearities located at approximately the -0.35 and 0.2 V threshold voltages. In particular, there appear to be two regions of memristor operation between -0.35 to -0.05 V and between 0.05 and 0.2 V which the linear memristor model fails to predict accurately.

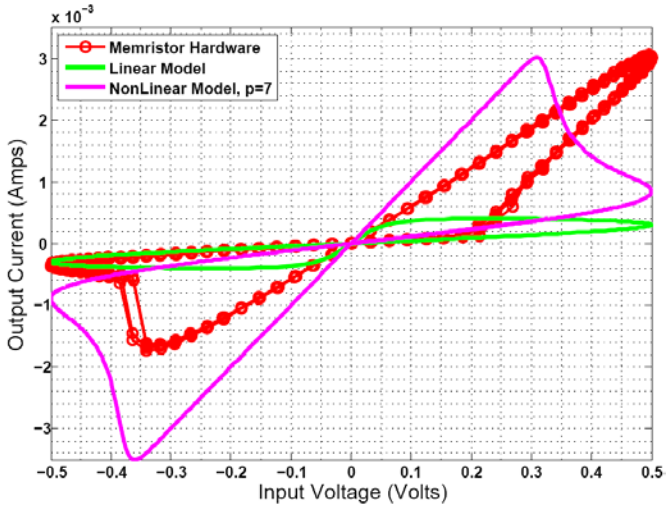


Figure 9. Memristor hardware and model fit correlation

Similarly, Figure 9 describes the nonlinear theoretical I-V curves for a memristor device with (realistic) dopant drift modeled by window functions obtained by Joglekar et. al [16], nonlinear model curve in magenta color. From the figure, it is clear that the nonlinear model results do not accurately describe the DC electrical Lissajous I-V characteristic behavior of our experimental physical memristor device hardware. For the nonlinear model, if we consider the memristor regions of operation where the nonlinear behavior dominates below or above the threshold voltages, -0.35 and 0.2 V respectively, both hardware and linear and nonlinear models exhibit completely different characteristic behaviors as displayed in Figure 4 respectively. Given the lack of experimental and model correlation for both linear and nonlinear memristor models, we have developed a simple empirical memristor compact model that we can employ to accurately capture the DC electrical Lissajous I-V curve characteristic behavior of memristor devices.

One important characteristic of memristor devices is the fact that their present behavior is dependent on their past state. Therefore, our simple compact model electronic characteristic behavior is dependent on the previous memristor state requiring initial conditions. For example, $R_{mem}(t=T)$ represents the state of the memristor device at an initial time T, and R_{on} and R_{off} represent final states of the memristor device. Assuming that the memristor device initial state corresponds to $R_{mem}(t=0)=R_{on}$ and that in time an input potential voltage greater than Vth is present across the device, we can describe the behavior of the memristor device as follows:

$$R_{mem}(t) = \begin{cases} R_{mem}(t-\Delta t) - \Delta t K_{h1} e^{K_{h2}(V_{in}(t)-T_h)}, & if\ R_{mem}(t-\Delta t) < R_{on} \\ R_{on}, & else \end{cases} \quad (10)$$

where $\Delta t=1e^{-3/f}$ corresponds to the minimum integral time step between memresistance observations, f is the frequency of the sinusoidal input voltage, T_h corresponds to the threshold voltage required to enter the nonlinear region from the off region and K_{h1} and K_{h2} correspond to fitting parameters used to capture the nonlinear effects characteristic of the memristor device.

On the other hand, if an input potential voltage lower than V_{tl} is present across the device, we can describe the behavior of the memristor device as follows:

$$R_{mem}(t) = \begin{cases} R\_mem(t-\Delta t) - \Delta t K_{l1} e^{K_{l2}(V_{in}(t)-T_l),} & if\ R_{mem}(t-\Delta t) > R_{off} \\ R_{off,} & else \end{cases} \quad (11)$$

where T_l corresponds to the threshold voltage required to enter the nonlinear region from the on region, K_{l1} and K_{l2} correspond to fitting parameters to capture the nonlinear effects characteristic of the memristor device.

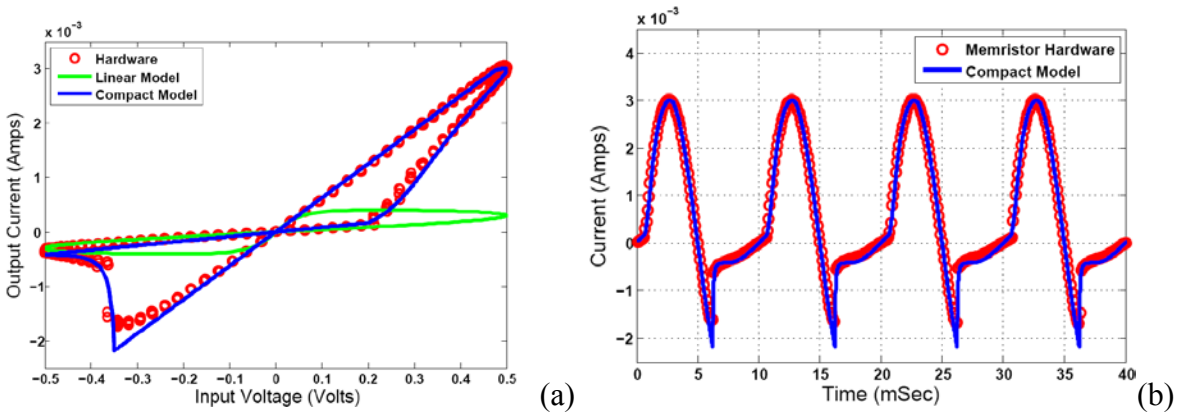


Figure 10. Memristor hardware versus model correlation (a) and memristor hardware fit versus compact model as function of time (b)

Otherwise, the state of the memristor device remains unchanged, and therefore the present state of the memristor equals that of its previous resistance state, $R_{mem}(t) = R_{mem}(t-\Delta t)$. It is important to highlight the diode-like behavior exhibited by our memristor devices driven by their internal highly nonlinear transport process. In fact, a closer look at the compact model equations (10) and (11) will reveal that our compact model equation resembles that of semiconductor diode devices from which we obtain the inspiration to develop our memristor compact model. Figure 10(a) shows the correlation between the hardware electrical characterization data and the memristor model's results for the linear memristor model (for reference) and our proposed memristor compact model. From Figure 10(a), we can observe that the compact model is able to describe all regions of DC electrical operation of the memristor device including the high and low conductivity regions and the nonlinear regions. The compact model fitting parameters to achieve the compact model fit displayed in Figure 10 correspond to R_{on}=160, R_{off}=1200, T_h=0.2, T_l=-0.35, K_{h1}=5.e6, K_{h2}=-20, K_{l1}=4e6, K_{l2}=20, an input sinusoidal voltage of 0.5 V amplitude and 100 Hz frequency and the initial condition that the state of the memristor = Roff. In addition, Figure 10(b) displays the time dependent compact model results versus the experimental results for the memristor device hardware showing a good model-to-hardware correlation in the time domain.

3.2.2 Operating in a Constrained Environment

Developing neuromorphic computing architectures is challenging no matter what environment and size platform one is dealing with in their research and development effort. The problem is only compounded when one seeks to implement advanced neuromorphic computing technology in a SWaP constrained platform such as a micro air vehicle (MAV) in a contested environment. A contested environment is a difficult place to operate if there is limited information about the location, access to global positioning satellite information (GPS) to aid in navigation is impeded, and communications links are unavailable for long periods of time. Operating in this kind of environment requires technology that will allow an unmanned system to have more autonomous capability. This is where neuromorphic computing and other bio-inspired technologies for SWaP constrained environments can play a key role.

One portion of this effort was spent obtaining a better understanding the requirements of a constrained environment, what can be done with existing technology, integration issues and technology gaps for on-board processing that exist. In order to conduct this research, since the interest is in developing advanced on-board processing capability not developing new robotics platforms and sensors, we relied heavily on commercial off-the-shelf robotics technology and hardware from the open source community. More specifically we used a DFRobot basic four wheel platform, Arduino Uno and Mega 2560 microcontroller boards, a motor shield that interfaced with the Arduino boards, and a variety of sensors. These sensors included: a "PING)))" ultrasonic sensor, Sharp infrared (IR) GP2Y0A21YK0F range finder, and a Maxbotix LV-MAXSONAR-EZ0 sonar range finder. Work progressed in increments starting with learning to program the Arduino boards, to integrating the Arduino boards with the robotic platform and getting initial movement, to incorporating a single sensor for obstacle avoidance, and on to the use of multiple sensors for simple navigation. Once the initial integration issues were overcome and basic functionality was achieved, more advanced concepts such as object tracking and the utilization of optical flow were pursued.

Object Tracking

One of the operational challenges for an unmanned system is to track and follow an object. In order to obtain firsthand knowledge of the issue of object tracking, a light tracking algorithm was used in conjunction with a vision chip to track a flashing light-emitting diode (LED). For this part of the effort, Tam 2 chips mounted on ArduEye Rox1 shield boards were obtained from Centeye, Inc. [18]. Figure 11 shows a mounted Tam with a pinhole lens. This vision chip is a 16 x 16 array of pixels that respond to the intensity of light striking them. Since it is flashing, the LED signal received by the vision chip can be easily distinguished from other light sources. In the present implementation, the LED flashes at a frequency of 180Hz. At this frequency, the LED appears to be continuously lit to the human eye. The tracking algorithm works by finding the position of the LED in the image at each time step and adjusting the rotation of the robot to center the LED horizontally in the image.

The first step of the light tracking algorithm is to sample the vision chip image. The image is sampled at 200 frames per second. This sampling rate results in the downshifting of the LED frequency to 20Hz. To reduce the necessary computation, the sum of each column is taken since the vertical position of the LED does not need to be known. Adding the pixels in the columns reduces the number of pixels to 1/16 of its original value and reduces the required computation by the same amount.

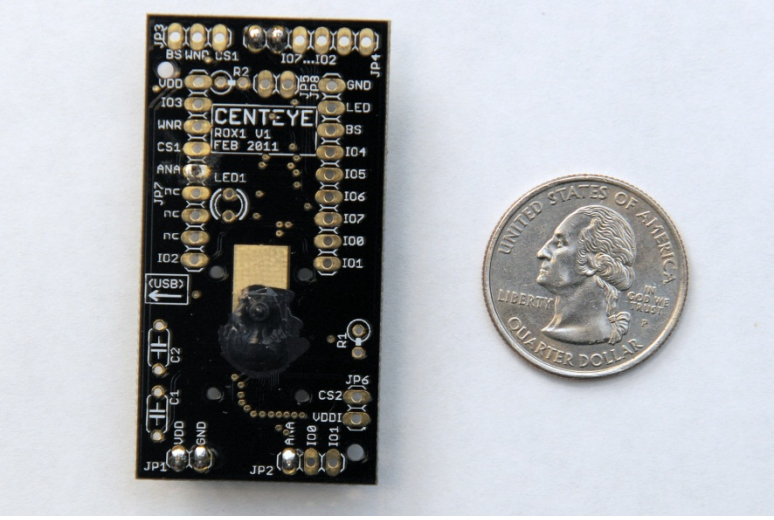


Figure 11. Centeye Tam 2 chip mounted on an ArduEye Rox1 shield board

After the 16 pixel values have been obtained, each of the 16 pixel values are band pass filtered with a center frequency of 20Hz to remove signals that are not the desired LED signal. The poles and zeros of the band pass filter transfer function are listed in Table 1. The magnitude of this transfer function is plotted in Figure 12.

After each pixel signal has been bandpass filtered, the value of each pixel is squared. The multiplication of the signal by itself in the time domain results in a convolution of the signal with itself in the frequency domain. The frequency response after squaring the signal is plotted in Figure 13.

Table 1. Band Pass Filter Poles and Zeros

Zeros	Poles
1	$0.94e^{j0.24\pi}$
-1	$0.94e^{-j0.24\pi}$
$e^{j0.1\pi}$	$0.93e^{j0.16\pi}$
$e^{-j0.1\pi}$	$0.93e^{-j0.16\pi}$
$e^{j0.3\pi}$	
$e^{-j0.3\pi}$	
$e^{j0.4\pi}$	
$e^{-j0.4\pi}$	
$e^{j0.6\pi}$	
$e^{-j0.6\pi}$	
$e^{j0.8\pi}$	
$e^{-j0.8\pi}$	

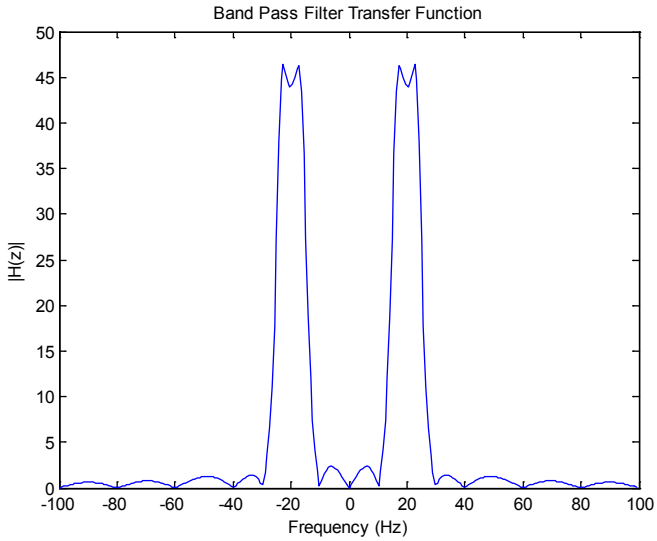


Figure 12. Band Pass Filter Magnitude Response

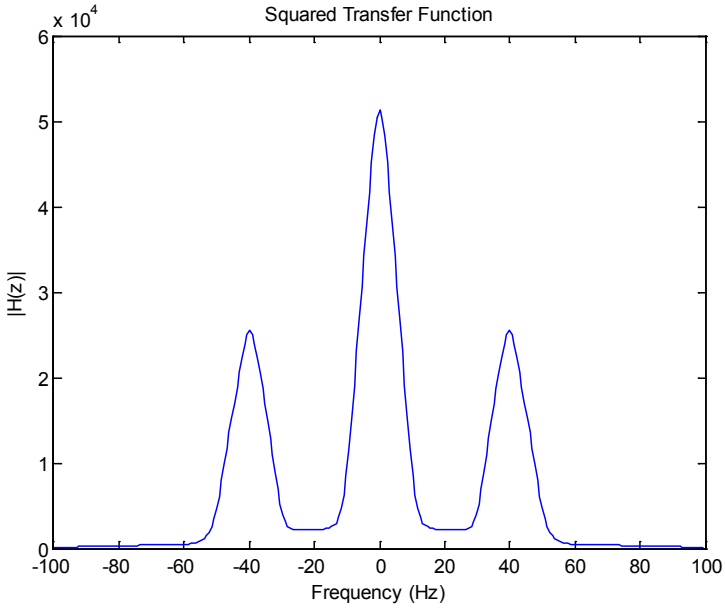


Figure 13. Frequency Spectrum of Squared Signal

The lobe centered at 0 Hz represents the intensity of the signal received by the pixel. This is the part of the signal that we are interested in, so the other two side lobes are removed using a low pass filter. Table 2 gives the poles and zeros for the low pass filter.

Table 2. Low Pass Filter Poles and Zeros

Zeros	Poles
-1	$0.9e^{j0.09\pi}$
$e^{j0.3\pi}$	$0.9e^{-j0.09\pi}$
$e^{-j0.3\pi}$	
$e^{j0.4\pi}$	
$e^{-j0.4\pi}$	
$e^{j0.5\pi}$	
$e^{-j0.5\pi}$	
$e^{j0.8\pi}$	
$e^{-j0.8\pi}$	

The magnitude of the frequency response of this transfer function is plotted in Figure 14. The filter was designed to increase the bandwidth of the center lobe of the signal and to provide large attenuation to the two side lobes. Increasing the bandwidth of the center lobe decreases the response time of the system so that the robot will react more quickly as the position of the LED changes. It also makes the system more susceptible to noise. A tradeoff is made between low noise and quick response time. The frequency response of the signal after low pass filtering is shown in Figure 15.

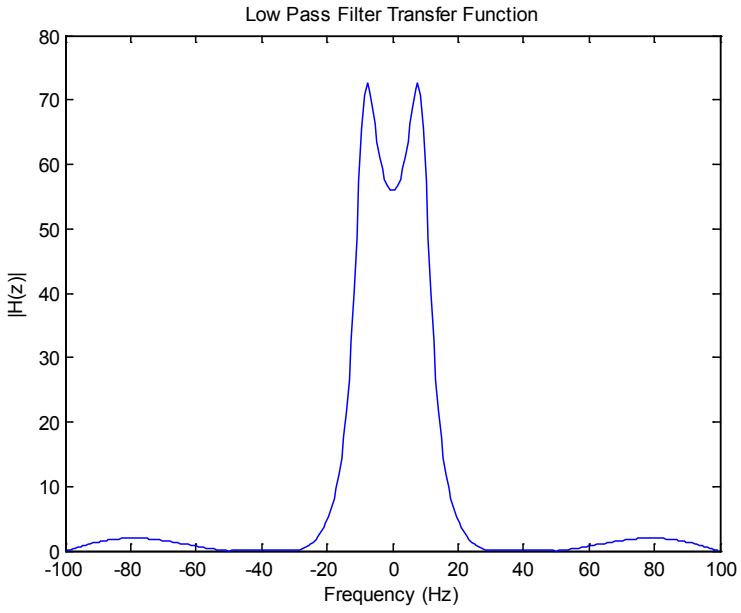


Figure 14. Magnitude Response of Low Pass Filter

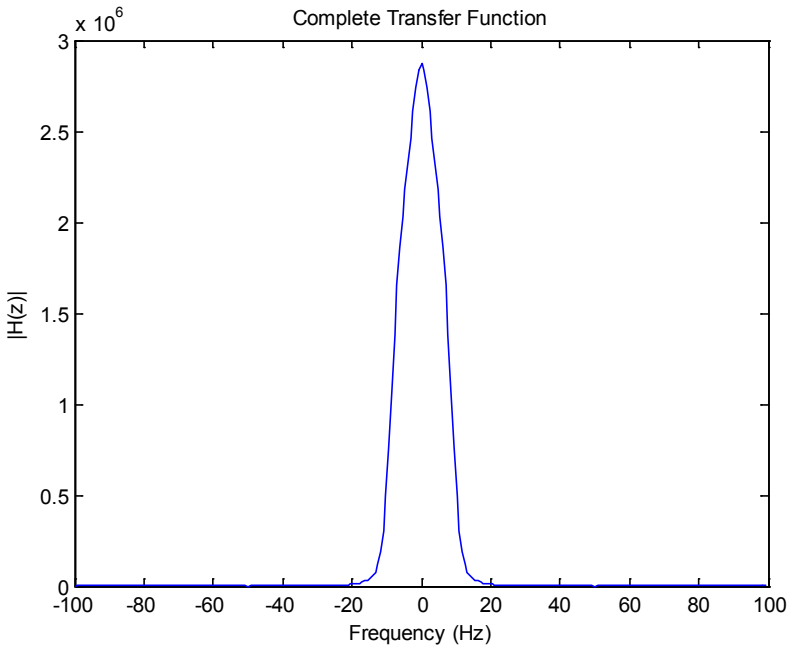


Figure 15. Frequency Spectrum after Low Pass Filtering

The signal at this point represents the intensity of the pixel. To determine the position of the LED, all 16 pixel intensity values are compared to each other and the maximum pixel represents the position of the LED.

To make the robot follow the LED, the robot is made to turn either left or right while it is moving forward so that the LED always remains centered in the image. An infrared (IR) rangefinder sensor placed in the front of the robot is used to stop the robot when it gets too close to the LED. Figure 16 shows an early picture of the Ardueye Shield attached to an Arduino Uno board prior to the mounting of the IR sensor.

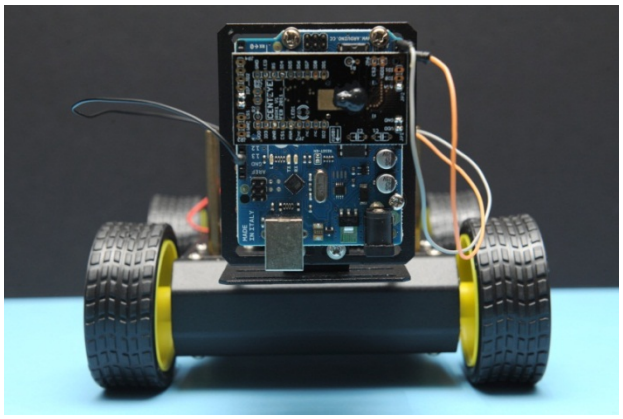


Figure 16. Ardueye Shield mounted to an Ardunio Uno board on a robotic platform

Optical Flow

Optical flow is a technique that measures the speed of objects picked up by a camera and can be used in various applications such as robot navigation. Two different methods of optical flow are discussed: the gradient method and feature tracking method. The gradient method is employed to detect when an object is in front of the robot. The principle idea of this method is when the robotic platform moves toward the object, objects closer to the robotic platform will move across the screen at a greater velocity than objects farther away. The feature tracking method is used to accurately turn the robot. As the robot turns, image features will move across the screen and the amount of rotation can be determined by how far the features move. The optical flow is calculated from images received by a vision chip that captures a 16 x 16 pixel image. The performance of using neural networks versus a defined algorithm to interpret the optical flow information for detection of an object is presented in between the explanation of the implementation of the gradient and feature tracking methods.

Optical Flow Gradient Method

The gradient method of optical flow calculation uses the spatial derivative and time derivative to determine the optical flow at each pixel in the image. The gradient method employs the equation given in [19],

$$v = \frac{-I_t}{I_x}, \tag{12}$$

where I_t is the time derivative of each pixel, I_x is the spatial derivative at each pixel, and v is the optical flow.

The first step in implementing this method is to read an image off of the vision chip. The pixels in each of the 16 columns of the image were summed to obtain an array of 1x16 vertical pixels. This is done since one only wants to find the optical flow in the horizontal direction. Next, pixel offsets are removed by high pass filtering the signal from each pixel in time. The high pass filter removes the DC offset of the signal. The DC offset of the pixels needs to be removed because differences in the DC offset of each pixel led to DC offsets in the spatial derivative which will create errors in the optical flow calculation. The high pass filter is implemented using:

$$Y[n] = 0.9\,Y[n-1] + X[n] - X[n-1], \tag{13}$$

where Y is the output of the filter, X is the input, and n is the current time step.

Next, to make the calculation of the spatial derivative more accurate, each frame is up-sampled by a factor of 4 so the 16 pixel horizontal array becomes a 64 pixel array. This is accomplished by repeating each pixel 4 times and then spatially low pass filtered the resulting array. The low pass filtering smoothes out the boundary between pixels. The low pass filter is implemented using the following equation:

$$Y[n] = 0.8\,Y[n-1] + X[n] + 0.64\,X[n-1] + 0.2548\,X[n-2], \tag{14}$$

After the initial processing of each frame, the optical flow is calculated using Equation 12. A problem with this equation is that as the derivative I_x approaches zero, the optical flow approaches infinity. Since the DC value of the pixels is removed, I_x is zero when there is no movement in the image. This equation will give a large value for the optical flow when there is no movement in the image. To remedy this, the $1/I_x$ term is approximated using the following exponential function:

$$\frac{1}{I_x} = \begin{cases} Ae^{-BI_x} & when\ I_x > 0 \\ -Ae^{BI_x} & when\ I_x < 0 \end{cases} \tag{15}$$

In this approximation, as I_x approaches zero, the output approaches +/-A.

After calculating the optical flow at each pixel, the absolute value of the optical flow values in the left and right half of the image are added up to get the total optical flow in the left half of the image and in the right half of the image. To determine when an object is in front of the robot, the sum of the two optical flow values are compared against a threshold. A total optical flow that is greater than the threshold means that there is an object in front of the robot. By comparing the left and right optical flow, it can be determined if the object is skewed to the left or right of the robot.

A second gradient method optical flow algorithm was created to attempt to get more performance. When the algorithm is first started, a calibration image is taken with the camera covered up so that no light enters the vision chip. This calibration image is stored in memory. Next, the algorithm loop starts and an image is taken from the vision chip. For each pixel in the image, the corresponding pixel in the calibration image is subtracted from the pixel value. Next, the top 4 rows and bottom 4 rows in the image are removed. This is done because we do not want the movement of the floor or ceiling to interfere with the output. Next, the pixels in each column of the image are summed resulting in a 16 integer array. Next, the optical flow for each pixel is calculated as follows:

$$Opflow\ =\ -100\ *\ (image1[i] - image2[i])/(image1[i+1] - image1[i])\,. \tag{16}$$

This equation is an approximation of $-I_t/I_x$ where I_t is the derivative of the pixel with respect to time and I_x is the derivative with respect to position. The equation is multiplied by 100 to increase the precision when the equation is implemented using integer arithmetic. When implemented in C, the order of operations is such that the multiplication is performed before the division reducing the loss of precision in the division.

Once the optical flow values are found, the total optical flow is found for the left side of the image and the right side of the image. This is done by taking the sum of optical flow values from index 0 to index 6 for the left side of the image and index 8 to index 14 for the right side. These optical flow values are low-pass filtered in time to reduce fluctuations in the output. This low pass filter is as follows:

$$Output\ =\ Output * \frac{19}{20} +\ Opflow. \tag{17}$$

Finally, the two optical flow values are divided by 20 so that the magnitude of each will fit into an 8 bit integer. This is done to simplify the transmission of the two optical flow values between the two microcontroller boards.

The following gives a list of the steps in the improved optical flow algorithm:

1. Cover camera and store the calibration image into memory.
2. Retrieve image
3. Subtract calibration image from image
4. Remove top and bottom 4 rows.
5. Add the pixel values of each column to get a 16 integer array.
6. Opflow = -100 * (image1[i]-image2[i])/(image1[i+1]-image1[i])
7. Output1 = Output1*19/20 + opflow(0:6)
8. Output2 = Output2*19/20 + opflow(8:14)
9. Output1 = Output1/20
10. Output2 = Output2/20
11. Go to 2

Software Neural Networks and Fixed Algorithms

Embedded systems typically have stringent SWAP constraints which can influence the selection of capability implementation strategies. A series of experiments were conducted that utilized either software ANN's or fixed algorithms to determine when the robot should stop in front of an object based on the optical flow information. In this case, the fixed algorithmic method was found to have several advantages with increased performance over the neural network approach. This was due in part to the memory and processor constraints of the platform and the limited training of the network.

In order to achieve good performance with the neural network, it is necessary to train the network with many training samples. These training samples consume a large amount of memory that is not needed with the algorithmic method. The number of training samples required is dependent on the complexity of the input data and test scenario, where increased numbers of input parameters corresponds to higher dimensional analysis. The neural network works by partitioning the input's space of the network into regions. If an input vector is in a region trained to be 1, the network outputs a 1 and if it is in a region trained as 0, the network outputs 0. The regions of the input space that the network partitions is programmed into the network by training it with samples of the input with known output. With a greater number of dimensions, a greater number of input samples are needed to accurately describe the desired partitioning of the input's space.

In this neural network approach, a feed forward neural network using the back propagation training algorithm was trained using all of the optical flow values for the 16 pixels. The neural network had 16 inputs and 1 output, one input for each optical flow value and the output to specify if there is an object or is not an object in front of the robot.

To reduce the number of training samples needed, the number of inputs on the network needs to be reduced as much as possible. Each input must be optimized to contain meaningful information. For determining when to stop the robot, the number of inputs could be reduced to one by adding all of the optical flow values. A one input neural network simply outputs a one if the input is greater or less than some threshold. Since the threshold is the only value that needs to be set, it is faster and more memory efficient to set this threshold manually as is done in the algorithmic approach. This approach is restricted to the simplest scenarios where data compression does not drastically affect task performance.

Another reason for the lower performance of the neural network in selecting when to stop the car is that the training samples must come from a wide range of different possible scenarios, which was not addressed in this simplified training approach. For example, if the robotic platform is trained to stop in front of a box with a certain orientation, it may very reliably stop in front of the box when the box is oriented in the position the robot was trained with but may fail completely if the box is rotated or if a different object is placed in front of the robot. Training samples must be taken from a large range of possible scenarios in order to accurately map out the entire desired region in the network's input space. The restricted training plan and memory for the storage of training data was the most significant aspect that limited ANN performance.

The use of neural networks in applications such as that discussed above can be beneficial in scenarios when an algorithm for selecting the correct output for a given input is unknown and a large amount of training data is available. It is generally beneficial to implement a fixed algorithm to perform such tasks in a predictable and stable environment, resulting in a simplified design that uses less computational resources than adaptable software based ANN schemes. However, the hardware implementation of neural networks has been shown to offer increased performance on a low power budget which is considered later in section 3.2.3.

Optical Flow Feature Tracking

The feature tracking optical flow algorithm was implemented to determine how far the robot is rotated. The method works by filtering out features from the image and tracking these features as the robot moves. In this case, the features are edges in the image.

This algorithm starts by creating a 16 pixel array as was done with the gradient method since we are still only interested in horizontal movement. An edge detection filter is then applied to the pixel array. This edge detection filter is a finite impulse response filter with the following impulse response: [1,0,-1]. Next, each pixel output from the edge detection filter is applied to a high pass filter to remove most of the DC component of the signal. This high pass filter is an infinite impulse response filter implemented using the following equation:

$$Y[n] = 0.9\,Y[n-1] + X[n] - 0.7\,X[n-1]. \tag{18}$$

After high pass filtering, the location of the maximum pixel, corresponding to one of the edges in the image, is found and this location is compared with the location of the maximum pixel in the previous image. If the difference between the two locations is 1 then this difference is equal to

the optical flow. If the difference between the two locations is greater than 1 then the optical flow is set to zero. This method assumes that the frame rate is high enough so that the features will move across the screen at a maximum rate of one pixel per frame. With this assumption, any difference between feature location greater than 1 means that the features being tracked in the two frames are two different features and using the difference in their location would give the wrong value for the optical flow.

A test was done to determine how well the turning algorithm worked. This test consisted of having the robot stop in front of a box and turn 90 degrees. The test was done at different light levels and with different orientations of the box. It was found that varying light levels did not have a noticeable effect on the performance of the algorithm while different orientations of the box did have a large effect. With some orientations the robot would repeatedly turn 90 degrees while with other orientations the robot would turn much more than 90 degrees. This lack of performance is most likely caused by the low resolution of the vision chip in relation to scene complexity, where the image that the vision chip captures may not have edges that the algorithm can track. To increase the performance of the system, a camera with a higher resolution would be needed that would be able to see more detail in the image which would allow more edges to be detected.

An alteration to the above algorithm was made to attempt to increase the accuracy. This new algorithm operates on the same principle but the details of its computation have changed. With this new algorithm, pixels in the columns are added up as before, and the mean value of the pixels in the image is subtracted from each pixel. During initialization a calibration image is created with all light blocked from entering the vision chip. This calibration image is used to removes pixel offsets. After each image is found and the average has been removed from each pixel, the calibration image is subtracted from each image to remove pixel offsets. Next, the edge detection filter is applied to the image as before. After edge detection, the local maximums and local minimums of pixel values are found for the image. This is done by assigning a 1 to a local maximum, -1 for a local minimum, and 0 if neither. After this, each local maximum and minimum point is compared with the same image point as well as the points beside it in the image from the previous time step to see if the minimum or maximum point has moved left or right. If the point has moved left, 1/N where N is the total number of maximum and minimums is subtracted from the optical flow value for that time step. If the point moves right, 1/N is added to the optical flow.

3.2.3 Hardware-based Artificial Neural Networks

The neuromorphic community was revitalized when, in 2008, memristive devices were brought to public attention by Hewlett Packard (HP), though devices possessing similar behavior, as predicted by Leon Chua in 1971 [20], had been observed since the 1970s [21-24]. These “memory resistors” possessed several attributes which make them effective hardware incarnations of biological synapses. First, these two-terminal devices function as variable resistors, satisfying the “learning” requirement seen in real synapses. Additionally, these devices are non-volatile. Unlike transistors, each memristive device’s state persists even in the absence of applied power. Lastly, these devices exhibit this behavior even in the nanoscale regime. For traditional computing, these properties could lead to instant-ON computers or non-volatile field-

programmable gate arrays (FPGAs). These devices potentially offer to neuromorphic circuit engineers the means to achieve the density, reconfigurability, and low power requirements needed to build analog neural circuitry. Memristive technology, however, must mature before it can be utilized in such designs. Memristor-built non-volatile memory is expected to be commercially available by 2014 [25], and HP predicts that memristive memory will eventually replace FLASH, solid state, and DRAM as a universal memory format [26]. Due to the disruptive nature of technology developments and to the exponential growth of technology as a whole, it is difficult to predict when, or if, memristive technology will enable neuromorphic thinking machines. Recent advances in silicon-based ANNs offer an alternative approach with many of the capabilities desired in future memristive systems but which are available now, CMOS compatible, and inexpensive.

Neuromorphic computing goals such as emulating mammalian brains prove daunting due to the processing power required to emulate all 4 million neurons of even a mouse's brain, much less that of a household cat, with approximately 300 million neurons. At the same time, extraordinary examples of pattern recognition and behavior are evident throughout the animal kingdom with significantly fewer neurons. For example, the roundworm, with 302 neurons and 8,000 synapses, can sense and track waterborne chemical signatures and navigate towards their locations [27]. We have found in the experimentation with hardware-based neural networks, presented in a later section of this report, that useful applications can be realized with relatively few active neurons. In fact, in one instance discussed later, only a single neuron was required to enable relevant change detection in a video surveillance system. Limited neuron approaches employed near the sensor may be used to reduce large data sets, saving critical transition bandwidth while reducing the burden on analysts and system operators.

Hardware-based ANNs are ideally suited for mobile or portable platforms with strictly limited SWaP resources. Such hardware implementations that operate without internal software offer improved efficiencies over traditional ANN software methods built on Von Neumann architectures, such as the ANN system described in section 3.2.2. The robotics industry (currently a $10 billion enterprise) is poised for exponential growth with an expected commercial market of over $15 billion by 2015 [28]. As an enabling technology, hardware ANNs will play a substantial role in the development of autonomous and semi-autonomous robots for use in industrial, commercial, and military markets. The demand for small, mobile, battery-powered systems will favor low-energy hybrid processing units consisting of both standard microcontrollers and hardware-based neural networks on the same chip. The parallel nature of these neural co-processors makes them superior over software techniques for SWaP restricted applications.

Our research focuses on the implementation of real-time ANN pattern recognition in platforms with severe SWAP constraints, such as micro air vehicles (MAVs), mobile sensor platforms, and pocket-sized robots. These restrictions practically rule out traditional software approaches which often run too slowly due to the inherent serial nature of von Neumann architectures or require high performance processing for operation. Hardware implementation of ANNs provides a reduced footprint with the additional benefit of massively parallel execution. While nano-enabled neuromorphic architectures are extremely promising, their realization will take time. Meanwhile, there exist commercially available technologies today that offer partial solutions. In

particular, parallel processing capabilities are afforded by FPGAs [29] and general-purpose computing on graphics processing units (GPGPUs). With the recent availability of application-specific integrated circuits (ASIC) based on zero instruction set computing (ZISC), not only is there an even greater reduction in footprint and power but also native support for massively parallel operation. It is technically feasible using current state-of-the-art fabrication techniques at the 22 nm node to manufacture ASIC ANN chips approaching 500,000 parallel neurons.

This research first examined the current state of memristive development with emphasis on architectural and fabrication challenges followed by a review the technical aspects of the CM1K ASIC chip and its ZISC operation. Next, two experiments demonstrating change detection and object recognition in live video feed will be presented. Lastly, the results of these experiments will be used to weigh in on two competing perception paradigms: few sensors/complex computations and many sensors/simple computation.

Challenges with Nano-enabled Neuromorphic Chips

A wide variety of technologies can be classified as "memristive devices" including resistive random access memory (ReRAM), phase change RAM (PCRAM), magnetoresistive RAM (MRAM), and spin-transfer torque MRAM (SST-RAM). While all these devices operate under different physical principles, they all possess two key attributes: 1) variable resistance and 2) non-volatility. Despite all the progress that has been made in memristive devices over the past several years, commercial memristive device products are still unavailable.

There are two critical tasks for successful memristive device integration with CMOS: manufacturability and usability. Concerning the former, the devices to be used must consist of materials that are permitted inside a CMOS foundry, which further restricts the materials allowed in the front end of line (FEOL) as compared to the back end of line (BEOL). All the processing steps needed to make the devices' structure must be scalable to fabricate devices en masse. Lastly, the devices must be all functionally identical (though some applications may actually exploit device non-uniformities). Part of the difficulty of manufacturing memristive devices is that the physics of device switching is not well understood at nanometer size scales. In particular, ReRAM (of which PCRAM is a subset) may be composed of binary metal oxides, chalcogenides, or perovskites, among other materials, and switch due to filament formation, vacancy migration, phase change, or other processes [30]. At these scales, small variations in the device size or material composition often have large effects upon subsequent device switching parameters.

However, because of this variety of materials and mechanisms, different device resistance values, switching voltages, and switching times are available to the circuit designer. When considering the appropriate device metrics of reliability and endurance that must be attained, one must first consider the intended use of the device. For von Neumann computing applications, if these devices are to replace Flash or SRAM, then endurance cycles of about 106 and write speeds of a couple tens of nanoseconds must be achieved, respectively. Even if memristive devices cannot meet these requirements, SWaP savings may still be achieved by strategically replacing some transistors in a circuit. For devices used in neuromorphic applications, the range of addressable resistance values and the operative voltages will be more critical than the write

speed. Because of these varied ends, there will likely be a variety of memristive device "flavors" available to the circuit designer in the future.

In the meantime, ZISC neural networks may be usable to partially replicate some of the anticipated advantages of a memristive device-based hardware neural network. In this way, learning algorithms and neural network structures may be further developed now and subsequently mapped onto memristive device technology if and when it becomes available.

ASIC Neural Networks

A fully parallel, silicon-based neural network chip (CM1K) developed by CogniMem Technologies Inc., based on IBM's earlier series of ZISC chips [31,32], was used in this work. The CM1K is configured with two available types of non-linear classifiers: a Radial Basis Function Network (RBF) and a K-Nearest Neighbor classifier (KNN). The chip possesses 1024 neurons, each with its own memory for trained signature storage and a processor for recognition and distance calculations. The memory within every neuron contains 256 elements, each with an 8-bit capacity for a total of 256 bytes of information per neuron. The identical neurons learn and respond to vector inputs in parallel while they incorporate information from all the trained neurons in the network through a bi-directional parallel neuron bus. Execution of the recognition logic is independent of the number of participating neurons, and multiple chips can be cascaded in parallel for scalable implementation. Figure 17 shows the general topology of such a restricted coulomb energy network. Each input node accepts a maximum of 256 elements (x_N), each with 8-bit resolution. These are fed in parallel to up to 1024 neurons. All recognition events are passed through to the output layer with the associated category and confidence level. CogniMem recently demonstrated a cascaded network of 100 chips with over 100,000 parallel neurons, all contained within 1/10 of a cubic foot and consuming less than 20 Watts of power yet performing at a level equivalent to 13.1 Teraops of pattern recognition performance [33]. Additional details regarding CM1K operation and architecture may be found in [34,35].

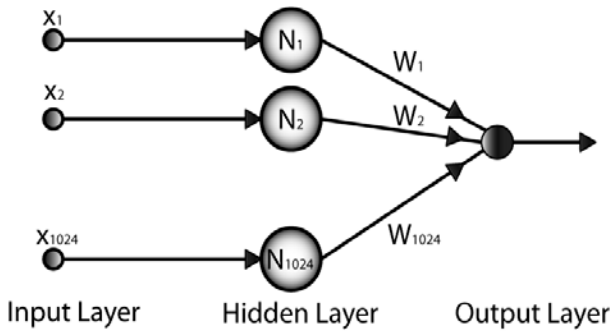
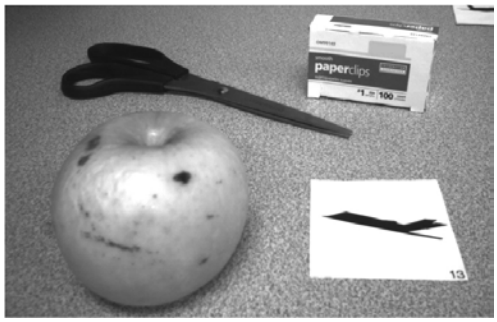


Figure 17. Neural Network Diagram

In such an architecture, the operational status of each neuron can be in one of three possible states: idle, ready-to-learn, and committed. The idle neurons are empty of knowledge but can be trained sequentially with the next neuron in the chain configured in the ready-to-learn state.

Once a neuron is trained it becomes committed and any pre-existing influence fields are adjusted to accommodate the new knowledge. During recognition, the input vector is passed to all the committed neurons in parallel, where it is compared to the stored vector or trained prototype. If the distance between the input vector and a neuron prototype falls within the influence field, the neuron "fires" generating local output signals consisting of fire flag, signal distance, and category type. In the case that no neurons fire, the input signal can be used to train the ready-to-learn neuron with the unrecognized signature. This provides the means for recognition and training to be accomplished simultaneously. A sample visual cue and grey scale image of the vector signature within a neuron's memory is given in Figure 18.

a. 

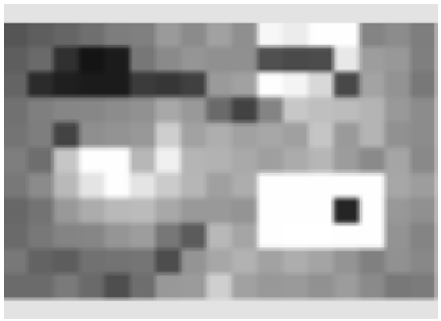
b. 

Figure 18. a) A captured video frame showing the subject and b) A pictorial representation of the neuron content trained with the above image

The program scans a 60 frames per second live video feed and compares the region of interest to those trained into the neuron's memory. Fine tuning of neuron sensitivity for a specific signature can be manually adjusted by adjusting the neuron active influence field or the distance from ideal, where a neuron will still recognize the target as a specific category. The distances can be calculated using one of two norms: the Manhattan method,

$$D_{Man} = \sum_{i=1}^{n} \left| V_i - P_i \right|, \tag{19}$$

where D_{Man} is the sum of the differences between n dimensional vector signatures V_i and P_i, or the L_{sup} method,

$$D_{L\sup} = Max \left| V_i - P_i \right|, \tag{20}$$

Manhattan distances were chosen for our design to emphasize the general differences between signatures with equal weights on all components. A neuron fires when the input vector lies within a specified distance, that is, falls within the influence field of a neuron in the decision space.

Experimentation with Minimum Neuron Requirements

Part 1 - Changes Detection in Live Video

We examined the problem of change detection in video surveillance with the intent of utilizing the least amount of processor resources. We started with the straightforward task of monitoring the entry point into a room for any activity. A single neuron was trained to recognize an image of the entry point. The neuron's influence field was manually adjusted to the sensitivity required to detect a subtle change to the field of view. The native monochromatic video feed to the CM1K was a progressive scan at 752x480 pixel resolution and 60 frames per second while the neuron memory was scaled down to 187 elements each with 8-bit depth. While maintaining the context of the broadcast vector, it required N+2 clock cycles to pass the input to the network. With the CM1K clocked at 27 MHz or 37 ns per clock cycle, the broadcast of our 187 element input vector took only 7.0 µs. A block diagram of the complete system configuration is shown in Figure 19.

Despite the low resolution of the stored prototype (187 bytes), the system reliably responded to changes in the camera's field of view (FOV) or to more localized changes in a region of interest within the FOV. In this case, the system was programmed to alert a human analyst of any change and to capture images of the disturbance for review. Data capture and transmission continued until the intrusion moves outside the sensor's FOV. It is important to mention that although we implemented our system under the control of a personal computer, the ZISC chip learns and recalls patterns without internal code or the need for constant external supervision. This simple implementation of a single neuron for video change detection proved to be very reliable over a 36 hour period of entryway monitoring identifying 26 events with zero false positives and zero missed occurrences.

Part 2 - Specific Target Recognition

Pattern recognition in complex scenes often plagues ANNs, since a subject's spatial orientation along with environmental variables such as lighting and background affect the system's ability to accurately perceive the target. These inconsistencies can be addressed using three basic techniques: increasing the number of neurons to account for variability, preprocessing the video to reduce variance, and controlling the environment or setting of the scan. While applications exist where control over situational effects can be adequately controlled by engineering the platform's environment [35], reducing the variance in video streams can be particularly challenging due to the inherent inclusion of erroneous and unpredictable background effects [36].

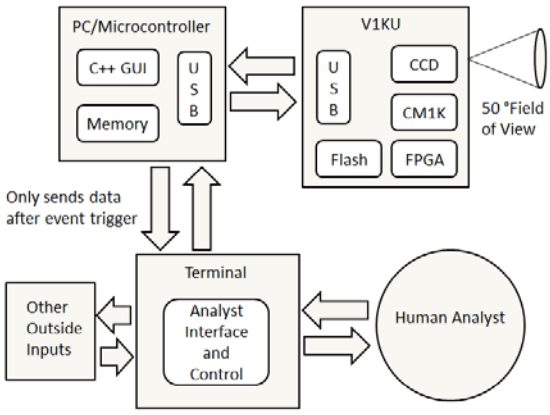


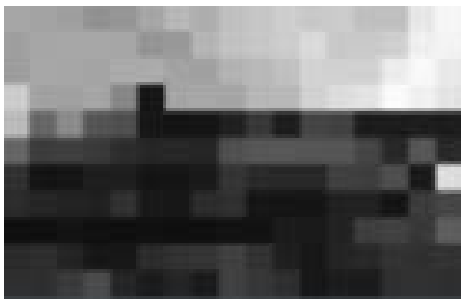
Figure 19. A block diagram depicting the system configuration

In this portion of the experiment, we tested the system's ability to detect specific targets in a controlled setting. Simulated vehicle traffic was monitored at the entrance point of a scaled model parking facility. A video camera monitored the incoming traffic (remote controlled cars), and the system alerted an attendant when either an unrecognized vehicle or a prespecified vehicle approached the gate. For this experiment, a pool of four vehicles under constant lighting was used. A single neuron was sufficient to identify a particular vehicle such as the one in Figure 20a. Figure 20b is a pictorial representation of the neuron content trained with the image in Figure 20a, while Figure 20c is a plot of a neuron's prototype vector. Since these neurons do not interpolate data, all flagged images had to be consistent with the orientation of the trained image.

With as few as one neuron per vehicle the system accurately identified each of the four vehicles and subsequently notified the analyst when a specifically flagged or unrecognized vehicle approached. Additional training was not required as long as the environmental aspects were held constant but improved reliability when environmental controls were lessoned.

The number of neurons required to distinguish N objects did not grow linearly but at a much faster rate, such that, 257 neurons were required to properly categorize 34 distinct targets under very controlled conditions. The realm of applications within a single processor's 1024 neuron capacity is significant but the need for additional performance through cascaded chips is required for complex relationships. Using this platform, we plan to construct a +100,000 parallel neuron network for additional research.

a) 

b) 

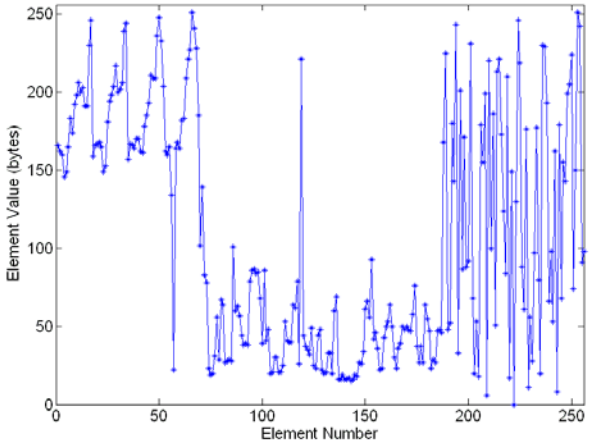
c) 

Figure 20. Images from experiment. a) Profile view of one of the vehicles recognized by one neuron, b) A pictorial representation of the neuron content trained with the above image, c) A plot of the neuron's prototype vector

4.0 RESULTS AND DISCUSSION

Numerous results were obtained in both the software and hardware focused research areas. The results are presented in the sections of the report which follow along with a discussion of their significance.

4.1 Software Focused Research

The Intelligent Text Recognition System (ITRS) achieved 2 critical milestones. First, it recognized 7000+ Chinese characters more quickly and precisely than its human counterpart. The 60x60 pixel recognition models can recognize characters possessing 40% pixel toggle with 80% success or with 20-pixel strike at 60% accuracy. Pixel toggle distortion at this magnitude is well beyond human-level cognition recognition capabilities. The ITRS can recognize the Chinese characters at a rate of 40-60 pages per second when instantiated on all the Graphics Processing Units (GPUs) on the AFRL/RI Condor Supercomputer. Secondly, ITRS incorporated semantic linguistic knowledge into its neuromorphic based sentence confabulation procedure. The Stanford Part-of-Speech (POS) tagger software was used to tag the existing training corpus and results indicate that tag-assisted confabulation improves sentence recognition 10%-55% of the time. The improvements vary with respect to the amount of sentence input noise, but the tag-

assisted confabulation shows clear improvement at all levels. Without tag-assisted semantic information, the overall success rate is expected to drop exponentially with respect to higher noise levels. Research continues under a complementary effort to incorporate confabulation into the Chinese character recognition aspect of ITRS. Furthermore, the utilization of POS for Chinese sentence recognition is still being explored. In the case of the English object character recognition with character occlusions, testing showed that ITRS out performed commercial object character recognition software in speed (40-60 pages/sec vs. 0.57 pages/sec), word accuracy (98.0% vs. 82.5%), and sentence accuracy (62-70% vs. 20%).

4.2 Hardware Focused Research

4.2.1 Memristive Device Based Technology

The memristor research outlined in this report consisted of physical modeling, compact modeling, device fabrication, testing, and application with the bulk of the device research focused on two memristor configurations, ion conductor chalcogenide-based memristor devices and Al/CuxO/Cu memristive devices. Physical and empirical models of these devices were created for MATLAB, HSPICE, & Verilog A environments. The models were compared to device testing results. It has been shown that memristive devices are a viable technology that may play a significant role in future neuromorphic and CMOS circuit design.

This work showed that a wide range of performance characteristics can be achieved using varied material and architectural designs, all demonstrating the characteristic pinched I-V hysteresis and non-volatility. Such variety makes standardization slow, but gives neuromorphic circuit designers expanded fabrication options. An in-house designed and fabricated reconfigurable logic circuit was used to demonstrate the utility of memristive behavior.

Although memristors are analog devices, both smooth and binary switching mechanisms were recorded. The smooth Lissajous curves typically documented in the chalcogenide based device testing more closely represents the traditional model of memristive behavior. These devices were very analog in function making them ideal for synaptic roles in neuromorphic circuits. Such analog systems allow the device physics to do the computation which more closely mimics the nonlinear dynamics of biological systems. But such complex circuits are difficult to reliably design and lack reprogrammability limiting their applications. Such subthreshold analog circuits are additionally sensitive to device variations, which can result in reduced bit resolutions after digital conversions. Fortunately many of the applications for neuromorphic systems do not require such exactness and are designed to solve the fuzzy computational problems too ambiguous for traditional processing. The Al/CuxO/Cu devices switched in a nearly binary fashion, providing a strikingly different Lissajous curve than the chalcogenides. There are several practical benefits of binary systems that are well known. Chief among these benefits is their insensitivity to noise followed by comparatively easy integration into current digital systems. Binary memristors are natural candidates for memory applications in nano-matrix

architectures. The most probable near term realizations of large scale neuromorphic circuits will continue to use both analog and digital circuitry in hybrid CMOS designs.

4.2.2 Operating in a Constrained Environment

The initial look at operating in a constrained environment provided some insight into issues that need to be addressed when trying to develop new technology for on-board processing with severe SWaP limitations. The integration of the right sensors with a microcontroller will provide basic functionality such as avoiding obstacles and navigating along a wall without direct human contact. Key to this work was having enough accessible input/output (I/O) points to bring all of the necessary data together. The Arduino Mega2560, with a 16MHz Atmega 2560 processor, was used as the primary board on the robotic platform carrying multiple sensors since it has 54 digital I/O pins, 14 pulse width modulation (PWM) pins, 16 analog inputs and four hardware serial ports. [37]. The Arduino Uno, with a 16 MHz Atmega 328 processor, on the other hand only has 14 digital I/O pins, six pulse PWM pins, 16 analog inputs and one hardware serial ports. While the Uno was more than capable of running the Tam2 on the Ardueye shield, there was difficulty using it with a motor shield and several different sensors due to accessing its smaller number of I/O pins. While this work with navigating a robotic platform is relatively simple compared to what will be needed to operate a small autonomous system in a contested environment, it was informative. The code for the target tracking research was separated into two programs. The first program was loaded onto the Arduino Uno board which was interfaced to the vision chip. This program performs all of the signal processing done to the image. The Uno board program contained 101 lines of code excluding comments of which only 4 lines actually perform the signal processing. The other lines were used to retrieve information from the vision chip, send processed information to the Arduino Mega2560 board, and for initializing the microcontroller. After processing the image, the LED position in the image was transmitted by Inter-Integrated Circuit (I2C) to the Mega2560 board. The second program was loaded onto the Arduino Mega2560 board which has the interface for controlling the robot drive motors. This program receives information from the Uno board and adjusts the speed of the wheels in the appropriate manner so that the LED is positioned in the center of the image. The Mega2560 board program contained 107 lines of code excluding comments. Most of the lines of code were for initializing and controlling the motors, the rest were used for receiving information from the Uno board and deciding how to control the motors depending on the information received. It will be interesting to see, in future effort to actually build and field small, autonomous, mobile systems, how many lines of code are required for the system to fulfill its mission and to process information closer to the sensor. Fancy and complex concepts do not always mean a better solution.

It has become more apparent, after this initial work and reviewing other published research, that autonomous operations in contested environments will require more on-board processing horse power. This requirement cannot be addressed by just using any chip with more processing capability especially if it requires much more power to run. One will not be able to just port over code from legacy systems. The challenge lies in determining the right mix of processing power and energy efficiency for a system. That is why other approaches, such as those that are bio-inspired such as vision chips and optical flow, need to be examined more closely to determine their merit and best way to use them. Dr. Geoffrey Barrows, in his dissertation for his doctorate,

for instance, acknowledged that, "The main challenge of using optic flow for navigation is that the computation of optic flow is known to be a CPU intensive problem."[19] If a fly can out maneuver a F-35 fighter in the low Reynolds number flight regime [4], than there are non-traditional means of computing that already exist in nature that need to explored and exploited. Traditional solutions will not get us to the computing efficiencies needed to operate small, autonomous, mobile platforms in a contested environment. At the time this work was conducted there was not a good guide for estimating, or a standard process for determining, the computing need of a small unmanned system. Researchers exploring navigation theories, mechanisms of bio-inspired flight, and fabrication of small vehicles tend to work with what they have worked with in the past, or that which was easy to obtain for their project that had adequate computing power. Relatively little effort has been made to address the on-board computing needs.

4.2.3 Hardware-based Artificial Neural Networks

The two simplified tests conducted under this effort illustrate that useful ANN systems can be designed to operate with extremely limited resources for use in SWAP constrained platforms. A nonlinear relationship was found between the number of prototype categories and the number of neurons required for identification, highlighting the need for increased density, low-power chip designs.

SRAM currently serves as the neuron's memory in this integrated circuit which consumes a significant amount of wafer real-estate and requires substantial energy resources for periodic refreshing. SRAM typically uses 6 transistors for storage and control of a single bit of CMOS memory. Additionally, since all the neuron knowledge is presently stored in SRAM, it must be saved off-chip in Flash when power is removed. Such requirements severely reduce the achievable density in SRAM-based neural networks. As mentioned previously, memristive devices hold great potential for the development of ultra-high density memory with reduced resource requirements. The non-volatile nature of memristive memory could lead to super-efficient ANNs that lay dormant without consuming power, ready to instantly respond when needed. By selectively replacing certain SRAM and DRAM transistors with CMOS-compatible memristive devices, ANNs can subsequently achieve increased density, reduced power, and instant-ON capabilities.

As was shown in both of the previous examples, a single neuron in this system was sufficient to achieve reasonable discrimination for pattern recognition. Coarse resolution in trained neurons may not necessarily be something that must be improved upon. In general, modern environmental processing relies heavily upon a small number of high resolution sensors and a computer capable of solving complex differential equations in real time. It is very unlikely that flying insects are performing such computationally intensive sensor processing in their brains. Rather, such biological systems perform simple computations using measurements from numerous crude sensors. This is called the sensor-rich feedback control paradigm. For example, it is thought that a fly determines a global vector representing how the fly is moving with respect to its environment. From observed vector patterns, select neurons fire when preferable flight directions are identified [4]. Extending this many sensors/limited computation paradigm to autonomous systems is a natural progression towards bio-inspired.

5.0 CONCLUSIONS

This neuromorphic computing research and development effort explored the design and implementation of computationally intelligent computer architectures and high performance computer software algorithms. The research concentrated on the design of mathematical models, algorithms, computing architectures, and computational efficiencies for the advancement of neuromorphic computing and neuroprocessors where the development of such systems draws from many fields.

The software focused research in neuromorphic computing demonstrated the combination of computational power with human level cognitive functionality. This sets the stage to create a system that can increase Department of Defense (DoD) operators and analysts ability to process textual data in depth, drawing more relationships among entities of interest. The neuromorphic software research is an enabling technology that will give cognitive functionality to the hardware.

The hardware focused research has evolved into a two prong approach to develop neuromorphic computing hardware for Air Force systems. One approach seeks to exploit memristive-based technologies for far-term concepts. As stated earlier, the memristor research outlined in this report consisted of physical modeling, compact modeling, device fabrication, testing, and application. A considerable amount of work was accomplished in this area under this effort, and in conjunction with AFOSR investment, leading to several papers and patents which can be found in the appendix. This is a very promising technology that will require more investment to develop the computer architectures which can exploit the technology for many Air Force applications such as autonomous operations in contested environments and enhanced on-board processing close to the sensor. While the memristive-based technologies mature, the second approach seeks to utilize CogniMem Technologies' CM1K-based hardware to explore system and architectural issues for more near-term solutions along with guiding memristive-based technology development.

A fully parallel, silicon-based ANN was used to monitor video data. In this work, change detection and simple object recognition were demonstrated with reduced neuron numbers utilizing only a few, or in some cases one, neuron per category. This simplified approach was used to validate the utility of few neuron networks for use in applications that necessitate severe SWaP restrictions. The limited resource requirements and massively parallel nature of hardware-based ANNs make them superior to many software approaches in such resource limited systems, such as MAVs, mobile sensor platforms, and pocket-sized robots. These fully CMOS compatible designs will likely play a substantial role in the development of semi-autonomous robotic platforms. Configurations having multitudes of crude sensors connected to layered ANNs will more closely emulate the structure of biological systems and may outperform systems which rely upon brute forcing complex equations upon data from a few high resolution sensors.

In the end progress was made on both the hardware and software side of neuromorphic computing research. The stage has been set for creating the future agile information systems with more capability for reasoning and perception that will allow the command chain to make informed decisions quicker than our adversaries.

6.0 REFERENCES

[1] Singer, P. W., **Wired for War: The Robotics Revolution and Conflict in the 21st Century**, The Penguin Press, New York, NY, 2009.

[2] Dahm, W. J. A., "Report on Technology Horizons: A Vision for Air Force Science & Technology During 2010-2030," URL: http://www.af.mil/shared/media/document/AFD-101130-062.pdf, May 15, 2010. Accessed March 29, 2013.

[3] Misra, J. and Saha, I., "Artificial neural networks in hardware: A survey of two decades of progress, *Neurocomputing*, **74**, 2010, pp. 239-255.

[4] Zbikowski, R.,"Fly like a fly," *IEEE Spectrum*, **42**(11), 2005, p 46-51.

[5] Menzel, R. and Giurfa, M. "Cognitive architecture of a mini-brain: the honeybee," *TRENDS in Cognitive Science*, **5** (2), 2001, pp. 62-71.

[6] Wysocki, B., McDonald, N., Thiem, C., Rose, G. and Gomez, M. "Hardware-based Computational Intelligence", submitted for Invited Book Chapter - Neuromorphic Information Processing, Springer Berlin/Heidelberg, expected publication summer 2013.

[7] Bi, G. and Poo, M., "Synaptic Modification by Correlated Activity: Hebb's Postulate Revisited," *Annual Review of Neuroscience*, **24**,2001, pp. 139–66.

[8] Lawrence, J. and Luedeking, S., **Introduction to Neural Networks**, California Scientific Software, Grass Valley, CA 1991.

[9] Pino, R., Patent US8275728. URL: http://patft.uspto.gov/netacgi/nph-Parser?Sect2=PTO1&Sect2=HITOFF&p=1&u=/netahtml/PTO/search-bool.html&r=1&f=G&l=50&d=PALL&RefSrch=yes&Query=PN/8275728. Accessed March 29, 2013.

[10] Pino, R., Bohl, J., McDonald, N., Wysocki, B., Rozwood, P., Campbell, K., Oblea, A., and Timilsina, A., "Compact method for modeling and simulation of memristor devices: Ion conductor chalcogenide-based memristor devices," *Nanoscale Architectures (NANOARCH), 2010 IEEE/ACM International Symposium on* , vol., no., pp.1,4, 17-18 June 2010.

[11] Chua, L., "Memristor - The Missing Circuit Element," *IEEE Transactions on Circuits Theory*, **18** (5), 1971, pp. 507–519.

[12] Strukov, D., Snider, G., Stewart, D., and Williams, R., "The missing memristor found," *Nature,* **453**, 2008, pp. 80-83.

[13] Williams, R., "How We Found the Missing Memristor," *IEEE Spectrum*, **45**(12), 2008, pp. 28-35.

[14] Chua, L., and Kang, S., "Memristive Device and Systems," *Proceedings of IEEE*, **64**(2), 1976, pp. 209-223.

[15] Biolek, Z., Biolek, D., Biolková, V., "Spice Model of Memristor with Nonlinear Dopant Drift," *Radioengineering,* **18** (2), 2009, pp. 210-214.

[16] Joglekar, Y., and Wol, S., "The elusive memristor: properties of basic electrical circuits," *European Journal of Physics*, **30**, 2009, pp. 661–675.
[17] Campbell, K., and Anderson, C., "Phase-change memory devices with stacked Ge-chalcogenide/Sn-chalcogenide layers," *Microelectronics Journal*, **38**, 2007, pp. 52-59.
[18] Centeye, Inc. URL: http://centeye.com/products/current-vision-chips-2/. (Accessed March 25, 2013).
[19] Barrows, G., Mixed-Mode VLSI Optic Flow Sensors for Micro Air Vehicles, Ph.D. Dissertation, Department of Electrical Engineering, University of Maryland at College Park, December 1999.
[20] Chua, L., "Memristor - the missing circuit element," *IEEE Trans. Circuit Theory,* **18**, 1971, pp. 507–519.
[21] Hirose Y., and Hirose, H., "Polarity-dependent memory switching and behaviour of Ag dendrite in Ag-photodoped amorphous As2S3 films," *J.Appl. Phys.*, **47**(6), 1976, p. 2767.
[22] Beck, A., Bednorz, J., Gerber, Ch., Rossel, C., and Widmer, D., "Reproducible switching effect in thin oxide films for memory applications," *Appl. Phys. Lett.*, **77**(1), 2000, p. 140.
[23] van der Sluis, P., "Non-volatile memory cells based on ZnxCd1-xS ferroelectric Schottky diodes", *Appl. Phys. Lett.*, **82**(23), 2003, p. 4089.
[24] Seo, S., Lee, M., "Reproducible resistance switching in polycrystalline NiO films," *Appl. Phys. Lett.*, **85**(23), 2004, p. 5655.
[25] Mellor, C., "HP's faster-than-flash memristor at least TWO years away, Plus: Storage boffins discuss photonic chip comms," *The Register*, URL: http://www.theregister.co.uk/2012/07/09/hp_memristor_and_photons/. Accessed September 19, 2012.
[26] Nickel, J., "Memristor Memory: A Fundamental Shift," 2011 *IEDM*, 2011.
[27] Shen K., and Bargmann, C., "The immunoglobin superfamily protein SYG-1 determines the location of specific synapses in C. Elegans", *Cell*, **112**(5), 2003, pp. 619–630.l
[28] ABI Research. "Personal Robots Are Here (and by 2015 They'll Be Worth $15 Billion)", URL: http://www.abiresearch.com/abiprdisplay.jsp?pressid=1023. Accessed January 3, 2012.
[29] Omondi, A., and Rajapakse, J., **FPGA Implementations of Neural Networks**, Springer, 2006.
[30] Pershin, Y., and Di Ventra, M., "Memory effects in complex materials and nanoscale systems," *Adv. Phys.*, **60**, 2011, pp. 145–227.
[31] Eide, A., Lindblad, T., Lindsey, C., Minerskjöld, M., Sekhniaidze, G., and Székely, G., "An implementation of the zero instruction set computer (ZISC036) on a PC/ISA-bus card", *WNN/FNN*, 1994.
[32] Dias, F., Antunes, A., Mota, A., "Artificial Neural Networks: a Review of Commercial Hardware", *Engineering Applications of Artificial Intelligence*, IFAC, **17**(8), 2004, pp. 945-952.
[33] Cognimem Technologies, Inc. *Cognimem Communique,* **1**(2), URL: http://general-vision.com/_docs/Newsletters/CogniMem%20Communique%27,%20Vol%201,%20Issue%202.pdf. January 10, 2012.
[34] Boulet, J., Louis, D., Godefroy, C., Steimle, A., Tannhof, P., and Paillet, G., Patent US5621863 URL: http://patft.uspto.gov/netacgi/nph-Parser?Sect2=PTO1&Sect2=HITOFF&p=1&u=/netahtml/PTO/search-bool.html&r=1&f=G&l=50&d=PALL&RefSrch=yes&Query=PN/5621863. Accessed March 29, 2013.

[35] Liu, Y., Wei, D., Zhang, N., and Zhao, M. , "Vehicle-license-plate recognition based on neural networks," *Information and Automation (ICIA), 2011 IEEE International Conference on* , vol., no., 2011, pp.363-366.

[36] Leoputra, W., Tan, T., and Venkatesh, S., "Unified 2D-3D Video Scene Change Detection Framework for Mobile Camera Platforms," *11th Int. Conf. Control, Automation, Robotics and Vision*, 2010.

[37] Adafruit Industries. URL: http://learn.adafruit.com/system/assets/assets/000/006/316/original/Table.JPG?1364045818. Accessed 29 march 2013.

APPENDIX – Patents and Publications

Patents:

1. U. S. Patent No.7,902,857, “Reconfigurable electronic circuit,” by Dr. Robinson Pino. Issued March 8, 2011.

2. U. S. Patent No. 8,249,838, “Method and Apparatus for Modeling Memristor Devices,” by Dr. Robinson Pino and James Bohl. Issued August 21, 2012.

3. U. S. Patent No. 8,275,728, “Neuromorphic Computer," by Dr. Robinson Pino. Issued September 25, 2012.

4. U. S. Patent No. 8,274,312, “Self-Reconfigurable Memristor-Based Analog Resonant Computer,” by Dr. Robinson Pino and James Bohl. Issued September 25, 2012.

Publications:

1. Pino, R., Li, H., Chen, Y., Hu, M., and Liu, B., "Statistical memristor modeling and case study in neuromorphic computing," *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE* , vol., no., 2012, pp. 585-590.

2. Bishop, M., Moore, M., Burns, D., Pino, R., and Linderman, R., "Affordable emerging computer hardware for neuromorphic computing applications," *Neural Networks (IJCNN), The 2010 International Joint Conference on* , vol., no., 2010, pp. 1-5.

3. Yakopcic, C., Taha, T., Subramanyam, G., Pino, R., and Rogers, S., "Analysis of a memristor based 1T1M crossbar architecture," *Neural Networks (IJCNN), The 2011 International Joint Conference on*, vol., no., 2011, pp. 3243-3247.

4. Pino, R., Genello, G., Bishop, M., Moore, M., and Linderman, R., "Emerging neuromorphic computing architectures & enabling hardware for cognitive information processing applications," *Cognitive Information Processing (CIP), 2010 2nd International Workshop on*, vol., no., 2010, pp. 35-39.

5. McDonald, N., Pino, R., Rozwood, P., and Wysocki, B., "Analysis of dynamic linear and non-linear memristor device models for emerging neuromorphic computing hardware design," *Neural Networks (IJCNN), The 2010 International Joint Conference on*, vol., no., 2010, pp. 1-5.

6. Bi, X., Zhang, C., Li, H., Chen, Y., and Pino, R., "Spintronic memristor based temperature sensor design with CMOS current reference," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012*, vol., no., 2012, pp. 1301-1306.

7. Hu, M., Li, H., Chen, Y., Wang, X., and Pino, R., "Geometry variations analysis of TiO2 thin-film and spintronic memristors," *Design Automation Conference (ASP-DAC), 2011 16th Asia and South Pacific* , vol., no., 2011, pp. 25-30.

8. Chen, Y., Li, H., Chen, Y., and Pino, R., "3D-ICML: A 3D bipolar ReRAM design with interleaved complementary memory layers," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011* , vol., no., 2011, pp.1-4.

9. Chen, Y., Li, H., Zhang, W., and Pino, R., "3D-HIM: A 3D High-density Interleaved Memory for bipolar RRAM design," *Nanoscale Architectures (NANOARCH), 2011 IEEE/ACM International Symposium on*, vol., no., 2011, pp.59-64.

10. Rose, G., Pino, R., and Wu, Q., "Exploiting memristance for low-energy neuromorphic computing hardware," *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, vol., no., 2011, pp. 2942-2945.

11. Rose, G.S.; Rajendran, J.; Manem, H.; Karri, R; Pino, R.E., "Leveraging Memristive Systems in the Construction of Digital Logic Circuits," *Proceedings of the IEEE* , **100**(6), 2012, pp. 2033-2049.

12. Miao Hu; Hai Li; Pino, R.E., "Fast statistical model of TiO2 thin-film memristor and design implication," *Computer-Aided Design (ICCAD), 2011 IEEE/ACM International Conference on* , vol., no., 2011, pp. 345-352.

13. Pino, R.E.; Bohl, J.W.; McDonald, N.; Wysocki, B.; Rozwood, P.; Campbell, K.A.; Oblea, A.; Timilsina, A., "Compact method for modeling and simulation of memristor devices: Ion conductor chalcogenide-based memristor devices," *Nanoscale Architectures (NANOARCH), 2010 IEEE/ACM International Symposium on* , vol., no., 2010, pp.1-4.

14. Pino, R.E.; Moore, M.; Rogers, J.; Qing Wu, "A columnar V1/V2 visual cortex model and emulation using a PS3 cell-BE array," *Neural Networks (IJCNN), The 2011 International Joint Conference on* , vol., no., 2011, pp. 1667-1674.

15. Hui Wang; Hai Li; Pino, R.E., "Memristor-based synapse design and training scheme for neuromorphic computing architecture," *Neural Networks (IJCNN), The 2012 International Joint Conference on* , vol., no., 2012, pp.1-5.

16. Moore, M., Linderman, R., Bishop, M., and Pino, R., "A columnar primary visual cortex (V1) model emulation using a PS3 Cell-BE array," *Neural Networks (IJCNN), The 2010 International Joint Conference on* , vol., no., 2010, pp.1-8.

17. Rose, G., Pino, R., Wu, Q., "A low-power memristive neuromorphic circuit utilizing a global/local training mechanism," *Neural Networks (IJCNN), The 2011 International Joint Conference on* , vol., no., 2011, pp. 2080-2086.

18. Pino, R., Bohl, J., McDonald, N., Wysocki, B., Rozwood, P., Campbell, K., Oblea, A., and Timilsina, A., "Compact method for modeling and simulation of memristor devices: Ion conductor chalcogenide-based memristor devices," *Nanoscale Architectures (NANOARCH), 2010 IEEE/ACM International Symposium on* , vol., no., 2010, pp.1-4.

19. Qiu, Q., Wu, Q., Bishop, M., Pino, R., and Linderman, R., "A Parallel Neuromorphic Text Recognition System and Its Implementation on a Heterogeneous High-Performance Computing Cluster," *Computers, IEEE Transactions on*, **62**(5), 2013, pp. 886-899.

20. Yang, F., Qiu, Q., Bishop, M., and Wu, Q., "Tag-assisted sentence confabulation for intelligent text recognition," *Computational Intelligence for Security and Defence Applications (CISDA), 2012 IEEE Symposium on*, vol., no., 2012, pp. 1-7.

21. Qiu, Q., Wu, Q., and Burns, D., Moore, M., Pino, R., Bishop, M., and Linderman, R., "Confabulation based sentence completion for machine reading," *Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), 2011 IEEE Symposium on*, vol., no., 2011, pp. 1-8.

22. Bishop, M., Moore, M., Burns, D., Pino, R., and Linderman, R., "Affordable emerging computer hardware for neuromorphic computing applications," *Neural Networks (IJCNN), The 2010 International Joint Conference on*, vol., no., 2010, pp. 1-5.

23. Kozma, R., Pino, R., and Pazienza, G. (Eds.), **Advances in Neuromorphic Memristor Science and Applications**, Springer Science+Business Media, Dordecht, 2012.

24. Wysocki, B., McDonald, N., Thiem, C., Rose, G. and Gomez, M. "Hardware-based Computational Intelligence", submitted for Invited Book Chapter - Neuromorphic Information Processing, Springer Berlin/Heidelberg, expected publication summer 2013.

LIST OF ABBREVIATIONS AND ACRONYMS

AF	Air Force
ASIC	Application Specific Integrated Circuit
BEOL	Back End of Line
BSB	Brain-State-in-a-Box
DoD	Department of Defense
DRAM	Dynamic Random-Access Memory
FEOL	Front End of Line
FLASH	Flash memory
FOV	Field of View
FPGA	Field-Programmable Gate Array
GFLOPS	Giga FLoating-point Operations Per Second
GPGPUs	General-Purpose Graphics Processing Unit
GPU	Graphics Processing Unit
HP	Hewlett Packard
HPC	High Performance Computing
I/O	Input/Output
IR	Infrared
KNN	K-Nearest Neighbor Classifier
LED	Light-Emitting Diode
MAVs	Micro Air Vehicles
MRAM	Magnetoresistive Random-Access Memory
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor
NFET	N-Channel Metal–Oxide–Semiconductor Field-Effect Transistor
NLP	Natural Language Processing
PCs	Personal Computers
PCRAM	Phase Change Random-Access Memory
PFET	P-Channel Metal–Oxide–Semiconductor Field-Effect Transistor
POS	Part of Speech
PWM	Pulse Width Modulation
RBF	Radial Basis Function Network
ReRAM	Resistive Random-Access Memory
SI	Synapse Input
SO	Synapse Output
SPE	Synergistic Processing Element
SRAM	Static Random-Access Memory
SST-RAM	Spin-Transfer Torque Magnetoresistive Random-Access Memory
SWaP	Size, Weight and Power
TFLOPS	Tera FLoating-point Operations Per Second
ZISC	Zero Instruction Set Computing