Intelligent Vision Systems: Exploring the State-of-the-Art and Opportunities for the Future

Siddharth Advani*, Srinidhi Kestur*,‡, Vijaykrishnan Narayanan*
*The Pennsylvania State University, USA
‡Broadcom Corporation

Abstract— **Vision and Video applications are becoming ubiquitous in mobile and embedded systems. The advent of wearable devices which require capabilities for real-time video analytics and prolonged battery lifetimes is further driving the need for innovative system designs with low-power, reliability and high performance. Further, the increasing resolution of image sensors in these mobile systems places an increasing demand on both the memory storage as well as the computational power. Such stringent requirements have given rise to accelerator-rich architectures in system-on-chips, where the primary computational burden is handled by dedicated hardware accelerators.**

In this paper we explore existing Vision accelerators and analyze their architecture, performance and scalability for different datasets and applications. The applications evaluated in this work are neuro-biologically inspired algorithms for object detection, object recognition and activity recognition which are complex, compute-intensive and bandwidth-intensive. This paper further analyzes the reliability of such embedded vision systems in terms of robustness of performance and energy efficiency under different application scenarios. Specifically, this work discusses the opportunities to improve energy efficiency by minimizing DRAM refreshes and explores techniques to exploit algorithmic resilience to minimize power consumption while maintaining reliable system accuracy and performance.

I. Introduction

Many chip-makers are now earmarking a significant amount of research effort for vision-based processors. Texas Instruments offers a heterogenous multi-core DSP for real-time vision applications using their Keystone architecture. Recently Freescale Semiconductor unveiled a vision system-on-chip - S32V - for accident-free-cars. Camera-friendly wearable devices like Google Glass are demanding better power efficiencies, improved performance and more powerful capabilities from the underlying technologies.

In the context of real-time vision applications, single-class object detection is a highly computationally intensive task. To robustly detect an object in an image that may appear at arbitrary position and scale involves (1) extracting optimized features that aptly describe the object and (2) searching the image in a sliding window fashion for the presence of particular configurations of the features that are indicative of the object's presence. This exhaustive search is compounded by objects that exhibit high appearance variability in shape, color and size. But, for visual-assist systems, the ability to perform such a task is imperative. For example, in a visual driving assist system, an approaching vehicle or a passing pedestrian needs to be detected with minimal latency, minimum false positives, and maximum accuracy. On the other hand, a wearable visual prosthesis device needs to augment the visual cognition of the user in diverse and vastly unconstrained environments for extended periods of time.

In this paper, we focus on xyz. To augment the next generation of wearables, we lay emphasis on abc. The main contributions of this paper are:

- We survey the state-of-the-art.
- We exploit reliability.

The rest of this paper is organized as follows: In Section II, we provide an overview of vision-based architectures and the corresponding state-of-the-art. Section III describes a robust object recogntion pipeline. Finally, we conclude with Section IV.

II. Related Work

Due to the capacity of human vision systems for highly complex processing at very low power, many brain-inspired algorithms and architectures have been proposed to emulate the human visual cortex. [1], [2], [3]. Most works in this domain have focused mainly on enhancing the performance and energy efficiency of the computational fabrics and do not address the inefficiencies of the main memory system. The memory system contributes between 10-30% of the overall power of embedded video systems and mobile phones [4]. The increasing memory size in new generations of embedded systems and the use of stacked 3D architectures that increase on-chip temperatures have drawn increasing attention on reducing the memory refresh energy. Consequently, there have been sustained efforts to introduce new power-efficient techniques such as Low Power Auto Self Refresh, Temperature Controlled Refresh, Refresh Pausing, Fine Granularity Refresh and Data Bus Inversion in new memory standards such as DDR4 [5]. Tuning DRAM refresh based on the data characteristics has been proposed as early as 1998 [6]. Recently, a software approach, termed as *Flikker* was proposed that relies on the user to annotate critical and non-critical parts [7]. It also allows refresh rates to be different for critical and non-critical sections of the memory and conserves the refresh energy.

In this work, we focus on the unique opportunities provided by real-time embedded video analytics applications for reducing memory refresh energy. The analysis is based on a real-time image detection and recognition system that has been emulated on an FPGA based platform which detects objects of interest using an attention algorithm. These objects can then be picked up from subsequent frames and classified. This system is useful in a range of end applications such as unmanned air-vehicles, security cameras and aids for the visually-impaired.

In this paper, we make the following contributions with regard to reducing the memory refresh energy in such a system. leftmargin=*

- We identify that in streaming data, the lifetime of some parts of the data are significantly less than the refresh periods of a DRAM and disable refresh in these parts of the memory.
- We automatically recognize portions of an image as critical based on the saliency-recognition algorithms employed in vision algorithms and selectively refresh those portions. In contrast, to [7] which statically annotates portions of the code and partitions them into critical and non-critical regions, our automated system can exploit both data dependent and task dependent information to identify salient regions within a single image frame.
- In embedded systems, the lifetimes of streaming data are significantly less than the refresh periods of a DRAM. This enables us to disable refresh in these parts of the memory and eliminate refresh for portions that are guaranteed to be accessed within a specific time period. We propose the underlying architecture to support such a visual pipeline with an optimized memory sub-system.

III. Reliability

In this section, we discuss the key design points of the pipeline. Figure 1 illustrates the end-to-end system beginning with the PCIe host data interface.

IV. Conclusion

In this work, we showcase alpha,betta,gamma

Future work entails uvw.

Acknowledgements

This work is supported in part by NSF Expeditions: Visual Cortex on Silicon CCF 1317560. The work is also supported through infrastructure provided by NSF Award 1205618.

References

[1] A. Nere, A. Hashmi, and M. Lipasti, "Profiling Heterogeneous Multi-GPU Systems to Accelerate Cortically Inspired Learning Algorithms," in *IPDPS*, 2011.

[2] T. Chen, J. Wang, Y. Chen, and O. Temam, "DianNao : A Small-Footprint High-Throughput Accelerator for Ubiquitous Machine-Learning," in *AS-PLOS*, 2014.

[3] S. Kestur *et al.*, "Emulating Mammalian Vision on Reconfigurable Hardware," in *IEEE International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, April 2012.

[4] A. Carroll and G. Heiser, "An Analysis of Power Consumption in a Smartphone," in *Usenix Annual Technical Conference*, 2010.

[5] "JEDEC DDR3 and DDR4 SDRAM Standard," 2012.

[6] T. Ohsawa, K. Kai, and K. Murakami, "Optimizing the dram refresh count for merged DRAM/logic LSIs," in *ISLPED*, 1998.

[7] S. Liu, Pattabiraman K., T. Moscibroda, and B. Zorn, "Flikker: Saving DRAM Refresh-power through Critical Data Partitioning," in *ASLPOS*, 2011.

Fig. 1. System Architecture mapped to an FPGA