Signal Processing: *Image Communication* 25 (2010) 352–362

Contents lists available at ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

Algorithmic and software techniques for embedded vision on programmable processors

Branislav Kisačanin [a,*], Zoran Nikolić [b]

[a] *Texas Instruments, Inc., Dallas, TX, USA*
[b] *Texas Instruments, Inc., Houston, TX, USA*

ARTICLE INFO

Article history:
Received 19 November 2009
Accepted 24 February 2010

Keywords:
Real-time
Embedded
Vision
Programmable
DSP

ABSTRACT

In the last few years, programmable architectures centered around high-end DSP processors have emerged as the platform of choice for high-volume embedded vision applications, such as automotive safety and video surveillance. Their programmability inherently addresses the problems presented by the sheer diversity of vision algorithms. This paper provides an overview of high-impact algorithmic and software techniques for embedded vision applications implemented on programmable architectures and discusses several system-level issues. We provide a general discussion and practical examples for the following categories of algorithmic techniques: fast algorithms, reduced dimensionality and mathematical shortcuts. Additionally, we discuss the importance of software techniques such as the use of fixed-point arithmetic, reduced data transfers and cache-friendly programming. In our experience, each of these techniques is a key enabler for real-time embedded vision systems.

1. Introduction

This paper is an overview of high-impact algorithmic and software techniques we learned and discovered over years of developing embedded vision systems on programmable processors. While most of our experience has been with TI digital media processors, the techniques we present are general and applicable to other similar architectures.

1.1. Embedded vision

As the first decade of the 21st century is coming to a close, we can confidently say that computer vision has entered the mainstream in consumer applications. As far as we know, the first vision-based system to be introduced to the consumer market was the optical mouse. While the early models of the 1980s required specially patterned mouse pads, the more modern versions, developed in the late 1990s, were based on optical flow (as described by Horn [13]) and did not require special mouse pads. This major step forward was facilitated by advances in algorithms and VLSI technology, allowing for an embedded implementation of the optical flow algorithm.

The introduction of the optical mouse to the consumer market was soon followed by the EyeToy, an attachment to the PlayStation 2 game console, comprising a camera and computer vision software, allowing gamers to visually become a part of the game. Today, we are witnessing a growing number of consumer products based on vision, especially in the automotive safety and video surveillance domains, with new toys, domestic robots and medical and mobile devices on the horizon.

Very much like computer vision applications for industrial inspection got a special name—*machine vision*, these recent applications of computer vision in safety, security, entertainment, medicine and mobility are now commonly called *embedded vision*.

* Corresponding author.
E-mail addresses: b.kisacanin@ti.com (B. Kisačanin), nikolicz@ti.com (Z. Nikolić).

0923-5965/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.image.2010.02.003

1.2. Is there an optimal vision processing architecture?

Much research effort has been dedicated to elucidating the optimal processing platform for computer vision. As documented by a 1992 special issue of IEEE Computer [4], many researchers and developers optimized the architecture for their algorithms and systems, while keeping them as general as their application allowed. One of the lessons of this experience has been that the architectures are highly application-specific.

For example, one-of-a-kind systems for many research projects often did not have to optimize for size, weight and power consumption because their primary objective was performance. Such projects could afford powerful workstation computers and even specially built hardware for the most challenging vision blocks. At the other end of the spectrum, in consumer products, where the priorities are quite different, we find that the optical flow for the optical mouse has been implemented on an ASIC (application specific integrated circuit), a decision justified by the huge number of products.

This example illustrates how the specifics of the product can dominate the architecture decision process and sometimes even mandate the answer. Thus, the prospect of a universally optimal vision architecture remains open. Yet the question of which platform is "optimal" for a particular application is still of immediate importance. Unless there are circumstances mandating the architecture choice, such as the extremely high volume of the optical mouse, one needs to consider the available choices and weigh them against the market requirements.

1.3. The challenges of embedded vision

While specific requirements vary between embedded vision systems, there are several common problems to be solved before prototypes can become consumer products. Sidestepping the "intangible obstacles," discussed by Chai [24, Chapter 11], that are major factors influencing technology penetration into the consumer world, we focus here on the following technological problems:

- *Algorithm diversity*. Vision algorithms are extremely diverse and not standardized. While textbook descriptions of algorithms are a good starting point, developers often find it necessary to make algorithmic changes specific to their system.
- *Data bandwidth*. Large data sets (images and video streams) need to be transferred. Data transfers in embedded vision are mostly on the input side, but sometimes also on the output side, in applications in which the user needs to see the output.
- *Processing latency*. Images and video need to be processed in real-time, often using highly pixel-intensive algorithms.
- *Product scalability*. Consumer products often have a varying selection of features, defining different product levels, each offering a superset of the features found in the lower product levels.
- *Cost*. Consumer products are sensitive to the system cost.
- *Energy*. Consumer products are also sensitive to energy issues (power dissipation and heat management).

1.4. Comparison of software- and hardware-programmable approaches

Embedded vision developers often find their applications to be between the extreme cases of one-of-a-kind systems that can be implemented on workstations and high-volume products like optical mouse, that can be implemented as ASICs. In such cases the developers have a number of options to choose from, as discussed by Kölsch and Butner [24, Chapter 1]. Our experience shows that in cases in which a large number of different algorithms is employed, with each running for a relatively short time, the software programmable architectures have advantage over the hardware programmable approaches. In this paper we look at software programmable architectures and describe techniques at three different levels: algorithmic, software and system.

1.5. Programmable architectures for embedded vision

In the following we consider general properties of programmable architectures that make them a good answer to the challenges of embedded vision. In the rest of this paper we assume a very basic model, depicted in Fig. 1, consisting of a processor and external RAM (random access memory). The processor itself comprises a CPU (central processing unit), DMA (direct memory access) controller and internal RAM. A part or all of the internal memory can be configured as cache.

The external memory is much slower than the internal memory, so for imaging and vision algorithms that access the data in rectangular blocks, it is beneficial to use dual buffering. In dual buffering the DMA controller is used to transfer the data between external and internal RAM, while the CPU processes the data previously transferred by the DMA controller to another buffer in the internal RAM. For other algorithms that have less regular data accesses, it is beneficial to use the cache memory, which predictively loads the data from the external memory. When cache is successful in predicting the data access patterns, the data access latency is closer to that of the faster internal memory than to that of the slower external memory.

Programmable architectures centered around high-end DSP (digital signal processing) processors are often the platform of choice for embedded vision. They are often VLIW (very long instruction word) architectures, which means that they consist of multiple execution units that can be pipelined for maximum parallelism. Additionally, some of these execution units have SIMD (single instruction multiple data) extensions, for example a 32-bit multiplier can be used to do four 8-bit multiplications simultaneously. With these and other desirable features,