

An Improved Real-Time Miniaturized Embedded Stereo Vision System (MESVS-II)

Bahador Khaleghi, Siddhant Ahuja, and Q. M. Jonathan Wu

Computer Vision and Sensing Systems Laboratory, E.C.E. Department, University of Windsor
{khalegh, ahuja5, jwu}@uwindsor.ca

Abstract

In this paper we describe a fully integrated, real-time, miniaturized embedded stereo vision system (MESVS-II), which fits within 5x5cm and consumes very low power. This is a significant improvement over the original MESVS-I system in terms of performance, quality and accuracy of results. MESVS-II running at 600MHz per core, is capable of operating at up to 20 fps, which is twice as fast as MESVS-I, due to the efficient implementation of stereo-vision algorithms, improved memory and data management, in-place processing scheme, code optimization, and the pipelined-programming model that takes advantage of the dual-core architecture of the embedded processor. The firmware incorporates sub-sampling, rectification, pre-processing, matching, LRC (Left/Right Consistency) check and post-processing. As demonstrated by our experimental results, we have also enhanced the robustness of the stereo-matching engine to radiometric variations by choosing census transform over rank transform.

1. Introduction

Stereo vision systems take advantage of the fact that the depth of the objects in the scene can be inferred from the relative displacements, also called disparities, of the objects in the scene, when observed from two viewpoints, separated by a distance. It is one of the most heavily investigated areas of research in the field of computer vision. Even though many new algorithms [1,2] introduced in this field focus on the quality and accuracy of the disparity (depth) maps, their successful implementation depends on the complexity of the algorithm and the availability of the hardware platform, which has a direct impact on their suitability for real-time implementation.

Some of the examples of real-time, operational stereo vision systems in the literature include systems relying on FPGA [3], commodity graphics [4], PC [5], ASIC [6] and hybrids incorporating various processor types [7]. The main focus of majority of these systems is on the raw

performance, rather than on the size, quality and power consumption.

We have developed a fully integrated, real-time, miniaturized Embedded Stereo Vision System (MESVS-II) which is a second generation design (first generation product was MESVS-I system), providing a significant improvement in terms of the performance, quality and accuracy of the depth maps. Operating at 20fps, it is twice as fast as MESVS-I. It is also more robust in-terms of radiometric variations by incorporating census transform over rank transform (more results presented in section 2.4.3). The main objective of this system is to provide a small, efficient, real-time, intelligent, stereo-vision sensor that can be applied in a wide variety of imaging applications in real-world environments.

MESVS-II, shown in Figure 1, fits within a tiny package of 5x5cm, produces accurate and quality depth maps at 20fps while operating at 600MHz per core, and consumes very low power (700mA@3.3V).

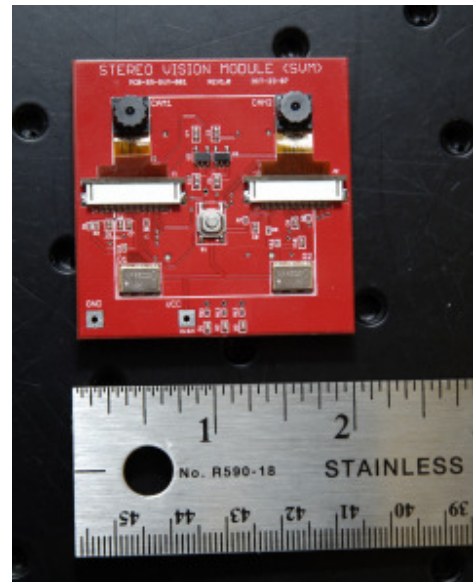


Figure 1. Prototype of the improved, real-time, Miniaturized Embedded Stereo Vision System (MESVS-II).

The stereo-matching engine performs sub-sampling, rectification, pre-processing, matching, LRC check, and post-processing. The performance, quality and accuracy was achieved by the efficient implementation of the algorithms, improved memory and data management, in-place processing scheme, code optimization and pipe-lined programming model, that leverages the benefits of dual-core architecture of our embedded vision processor.

The paper is organized as follows. Section 2 provides a description of the system design, including hardware implementation, memory architecture and software implementation (stereo-matching engine). A quantitative assessment of rank vs. census transform with respect to radiometric variations is presented in Section 2, along with details on how we achieved real time performance. The power characterization of the system is presented in Section 3 and the conclusion is provided in Section 4.

2. System description

Miniaturization of the stereo-vision system is accomplished not only by reducing the overall dimensions of the components used, but also by reducing the baseline, i.e. the distance between the camera's centers of projection. The size of baseline has a direct impact on the range of measurable objects in the scene (Horopter) and the related accuracy (range resolution). As can be seen from Equation 1, as the baseline b decreases, the range r decreases, whereas the range uncertainty Δr increases. Thus, there has to be a trade-off between the horopter and the associated accuracies for disparity values.

$$\Delta r = \left(\frac{r^2}{bf}\right)x\Delta N \quad (1)$$

where, f is the focal length, x denotes the pixel size and the change in disparity is denoted by ΔN .

To fit the system within 5x5cm, while having an acceptable measurable range and horopter, we have chosen the baseline to be 28mm, with horopter set to 5-35 and disparity range of 30, yielding a measurable range of about 15-100cm [8].

2.1. Processor selection

Selecting the desired processor for a computationally intense, stereo-vision application, is of critical importance, as it heavily influences the product cost, performance, and power consumption. There are many types of processors available in the market including PC CPU, Embedded RISC CPU, application processors (like DSPs¹), media processors (like ASIPs²), FPGAs (Field Programmable Gate Arrays), and ASSPs (Application Specific Signal

Processors). The selection criteria for choosing the proper processor type for the application of stereo-vision include:

- a. Performance considerations (like speed, memory handling, data buses, energy consumption, benchmarking results, etc.),
- b. Cost of Integration,
- c. Availability and roadmap,
- d. Development considerations (like single vs. multi-core, number of I/O ports, instruction set architecture, developer familiarity, compatibility, tools-compilers and profilers, support, etc.),
- e. Packaging requirements, and,
- f. Operating temperature range, among others.

Media processors like ASIPs provide higher performance than most DSPs and GPPs (General Purpose Processors) and have better support for video processing; however, they have complex programming models, higher developmental cost, and higher associated risk as their roadmaps are unclear. FPGAs can be reconfigured dynamically, offer architectural flexibility, high throughput and performance, all resulting in higher efficiency. However, their suitability for low-power, cost-sensitive, stereo-vision applications has not yet been proved. ASSPs incorporate one or more processor types that are well matched to the application, and thus offer excellent performance, and energy efficiency; however, ASSPs are often inflexible, have a sharp learning curve and require extensive tuning. Their roadmap is unclear and the benefits of low cost can only be realized when produced in mass-quantities. Application processors offer adequate performance, portability, energy efficiency, integration, and support for video-based applications; however, they are less powerful than other types of processors mentioned above [9].

Recently, a new family of Convergent processors (e.g. BlackFin by Analog Devices) has emerged that is ideal for advanced video processing applications, combining both MCU (Micro Controller Unit) and DSP functionality into a single device with a unified architecture, high clock speed, low power dissipation per unit of processing, smaller form factor and flexible programming model [10]. They function simultaneously as a 16-bit DSP and a 32-bit MCU while supporting both DMA (Direct Memory Access) and cache functionality. Embedded system programmers leverage the portability of the code written in C and try optimization approaches at the algorithm level and compiler level. However, in-order to achieve real-time performance, assembly language coding is required. Convergent processors allow the developers to create applications in C/C++, as the processor is optimized not only for computation on real-time multimedia data, but also for control tasks. The benefits include: best utilization of existing skill sets within a team, reduced time to market and lifecycle costs, higher processing speeds, and ease of

¹ DSP: Digital Signal Processor

² ASIP: Applications Specific Integrated Processors

maintenance. Thus, we have chosen the dual core, BlackFin processor (ADSP-BF561) as our processing platform for the stereo-vision system. More details on the hardware and memory architecture can be found in the following sections.

2.2. Hardware implementation

The system hardware consists of a state-of-the-art embedded processor ADSP-BF561, two tiny CMOS camera sensors, two Parallel Peripheral Interface (PPI), 64MB of SDRAM, 8MB of addressable flash memory, JTAG interface, and SPI port. Figure 2 shows a high-level block diagram of the system.

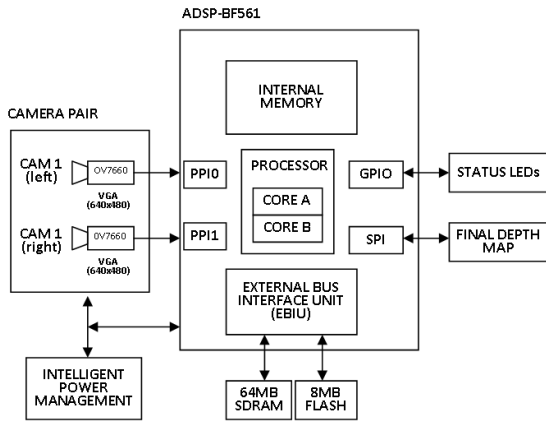


Figure 2. Block diagram of MESVS-II

ADSP-BF561 is one of the latest members of BlackFin family of embedded processors featuring a dual core processor, with each core capable of 1200 MMACs@600MHz (2400 MMACs total). ADSP-BF561 provides the best compromise between performance and size versus cost and power consumption [8]. It also possesses two flexible video ports used to capture stereo image pairs from CMOS cameras. Table 1 provides a summary of ADSP-BF561 specifications.

Clock Speed (MHz)	600MHz (per core)
MMACs (Max)	2400
RAM Memory (kBytes)	320
External Memory Bus	32bit
PPI ports	2
Core Voltage (V)	0.8-1.2
Packages	297-PBGA, CSP_BGA

Table 1. Summary of specifications for BlackFin processor

2.3. Memory architecture

Performance of an embedded system is directly dependent on how the memory and data is managed. BlackFin processors support a modified Harvard architecture along with a hierarchical memory structure. Figure 3 shows an overview of the memory architecture of our system. Each BlackFin processing core has access to the high-speed, high-performance, low-latency, Level 1 (L1) memory that typically operates at the processor speed. L1 memory is made up of 64KB of data memory

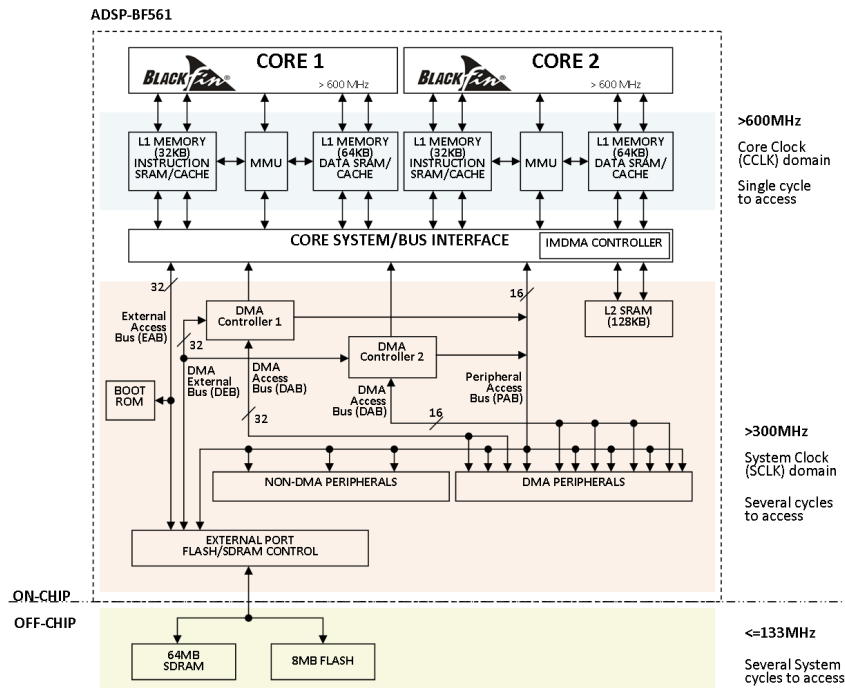


Figure 3. Memory architecture of MESVS-II.

and 32KB of instruction memory. Memory Management Unit (MMU) defines the properties of a given memory space and protects the system registers from unintended access. A unified Level 2 (L2) memory shared by both of the cores operates at approximately half of the core-clock speed, resulting in a higher latency compared to L1 memory. External memory also called Level 3 (L3) memory consists of 64MB of SDRAM and 8MB of flash memory. The system's firmware is burnt onto the flash memory. Although L3 memory is quite large, the access time is measured in System Clock Cycles (SCLK), which is usually much less than the CCLK rate.

Multiple DMA channels facilitate data movement between the peripherals and the memory systems, with no overhead on the processor. Extensive deployment of DMA functionality has allowed us to improve system performance and reduce the run-time of the matching algorithm, and rectification, as explained in section 2.5.

2.4. Stereo-Matching Engine

Calibration of the stereo vision system is done offline, using the Caltech camera calibration toolbox [11], to extract the intrinsic (e.g. focal length, pixel size, image center, non-linear radial distortion, etc.) and extrinsic parameters (e.g. rotation and translation) of the cameras. The various steps involved in the stereo-matching engine are outlined and explained in Figure 4(a-g).

2.4.1 Sub-sampling

The original images produced by the cameras are in VGA resolution and require 300KB of storage space per image. As explained in the previous section, the amount of on-chip, high speed, L1 memory is limited. In-order to fit the captured images into the L1 memory and ensure maximum performance, we sub-sample (see Figure 4a) the image pair into QQVGA (160x120) resolution using 2D DMA facility of BlackFin processor.

2.4.2 Rectification

The images are then rectified, to make the pairs of conjugate epipolar lines collinear and parallel to the horizontal image axis (see Figure 4b). This reduces the 2D correspondence problem to a simpler 1D search. Rectification step includes: back projection, distortion removal, and bi-linear interpolation.

2.4.3 Pre-processing

Pre-processing stage helps reduce the sensitivity of the matching algorithm to radiometric variations, due to local or global intensity changes, and noise. Non-parametric local transforms like rank and census transforms utilize the relative ordering of local intensity values [12]. Since they rely on a set of local comparisons, they are more stable, invariant to both gain and bias, and can tolerate

fractionalism [12]. Rank transform $R(P)$ of a pixel P , represents the number of pixels whose intensity $I(P')$ is less than the intensity of the central pixel $I(P)$ in a given square window $N(P)$, as shown in the equation 2 [12]

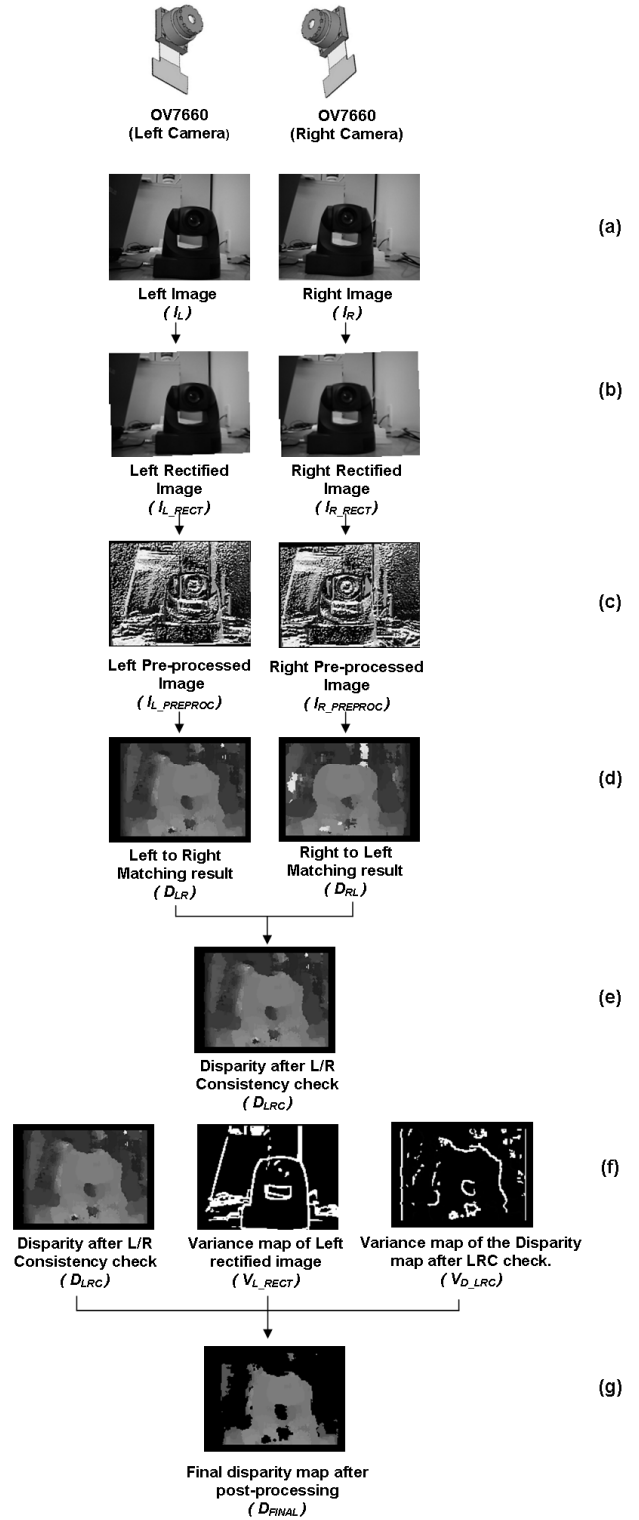


Figure 4. Overview of the stereo-matching engine.

below:

$$R(P) = |\{P' \in N(P) | I(P') < I(P)\}| \quad (2)$$

Census transform $R_T(P)$ represents the local intensity variation, expressed as a bit string made up of all the pixels in the neighborhood whose intensity value is less than that of the pixel P , given by equation 3 [12] below:

$$R_T(P) = \bigoplus_{[i,j] \in D} \xi(P, P + [i,j]) \quad (3)$$

where, \bigoplus is the concatenation operator on a set of pixels with a set of displacements D within a square window, and $\xi(P, P')$ is equal to 1 when $I(P') < I(P)$ and 0 otherwise.

Rank transform of two pixels is compared by utilizing the Sum of Absolute Differences (SAD) metric given by the following equation:

$$SAD(x, y) = \sum_{i,j=-n}^n |R_1(i, j) - R_2(x + i, y + j)| \quad (4)$$

where, the window size is $(2n + 1) \times (2n + 1)$.

Census transform of two pixels is compared by utilizing the Hamming distance metric given by the following equation:

$$\mathcal{H}(R_{T1}, R_{T2}) = \sum_{i=0}^{N-1} R_{T1}(i) \otimes R_{T2}(i) \quad (5)$$

where, \otimes is the Boolean exclusive-or operator.

Rank transform cannot distinguish between rotations and reflections, and has been shown to produce the same rank for variety of patterns [13]. It has also been shown that rank transform loses the information of spatial distribution of pixels surrounding the central pixel [14]. Census transform, on the other hand, preserves the local image structure [12], as it encodes it into a bit stream. It has been shown to produce better disparity results, due to lower incorrect matches as compared to rank transform [12, 14]. Both of the two transforms are suitable for fast hardware implementations [14].

According to the recent studies [15], in the presence of local radiometric variations, rank transform has been shown to outperform other methods such as LoG, NCC, and HMI for correlation based matching. However, a quantitative comparison of rank vs. census transform in the presence of global and local intensity changes (e.g., due to lighting and exposure differences) is lacking in the literature. In this paper, we have used the 2005 datasets taken from Middlebury stereo website [15], to compare rank and census transform under radiometric variations. The 6 datasets (Art, Books, Dolls, Laundry, Moebius, and Reindeer) contain images that were captured under three different exposures and lighting variations, resulting in 9 combinations of image pairs.

Figure 5 (a-b) show the average error percentage under

exposure and lighting variations, respectively, for both rank and census transforms, with a window size of 3x3. Figure 5 (c-d) shows the average error percentage under exposure and lighting variations for rank of size 5x5 and census of size 3x3. As can be seen from the experimental results, census transform of size 3x3 outperforms both rank of size 3x3 and 5x5 under radiometric variations.

To enhance the robustness with respect to radiometric variations, we have implemented census transform (size 3x3), along with the Hamming distance (see Figure 4c), as a mix of C and assembly code. Due to the availability of special in-built, single-cycle instructions (e.g. bit counting operation *Ones*, and bit-wise exclusive-or *Xor*), as a part of BlackFin's instruction set, we were able to speed-up the execution time of the pre-processing stage by approximately three orders of magnitude.

2.4.4 Matching

Stereo matching methods can be divided into local and global methods [16]. In case of local methods, disparity calculation is performed by aggregating local support using pixel values within a predetermined neighborhood of pixels. They make the implicit assumption that disparity values vary smoothly within the correlation window, therefore they fail in cases where this assumption is violated (such as depth discontinuity regions). On the other hand, most of the global methods enforce explicit constraints on the desired disparity map using an energy function. The final disparity map is obtained through a global optimization algorithm. Global methods usually outperform local matching algorithms; however, due to their higher computational complexity, they are rarely deployed in real-time stereo-vision systems. Some variants of global methods such as dynamic programming have been shown to be applicable in real-time vision system [7]; however, they only perform 1D optimization (just for the scan line) and have difficulty maintaining intra-scan line consistency throughout the disparity map, a problem also referred to as the streaking effect.

Local methods can be further categorized into block matching and feature matching methods. Feature matching methods extract viewpoint invariant features in the image such as edges, corners, and line segments and then try to match them, via pattern matching. Even though feature matching methods can be efficiently implemented, they are incapable of providing dense disparity maps.

Block matching methods typically calculate disparity maps through local image correlation over a window. They can produce dense disparity maps, and provide for fast implementations, and thus, they are suitable for real-time stereo vision applications. The correlation based matching algorithm uses hamming distance as the similarity measure (see Figure 4d). To speed-up processing, we have implemented three-levels of recursion namely horizontal, vertical and modular recursion [8].

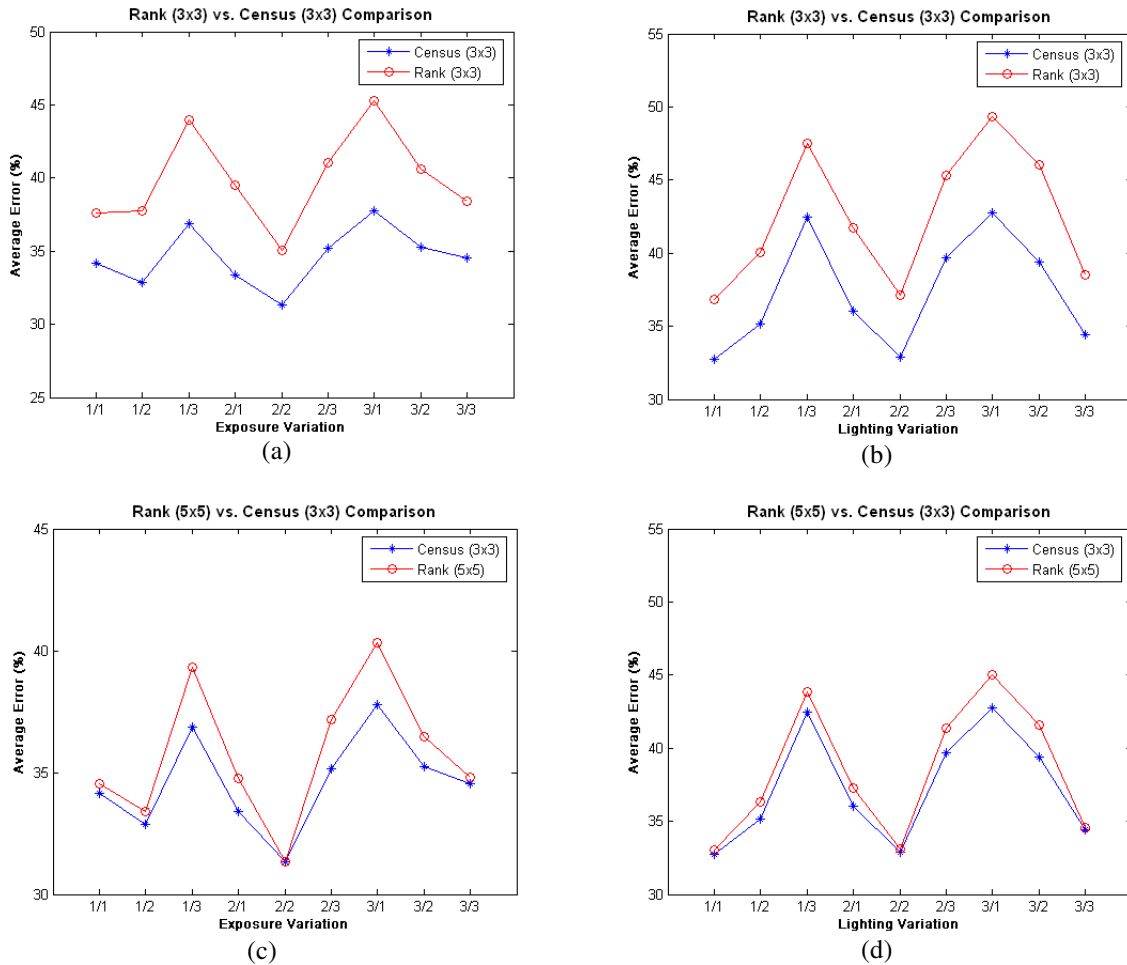


Figure 5. Rank vs. Census comparison for 3x3 left/right image combinations differing in exposure or lighting variations.

Furthermore, to treat the half-occluded regions in the scene, we have incorporated an efficient implementation of Left/Right consistency check algorithm (see Figure 4e).

2.4.5 Post-processing

The post processing step combines the variance map of the target image, variance map of the disparity map and the disparity map itself, to detect and remove erroneous disparities caused by texture-less and depth discontinuous regions in the scene. If the pixel of interest lacks enough texture, and the estimated disparity value is not in agreement with the neighboring pixels (measured by comparing variation of disparity values over a local neighborhood against a threshold), the estimated disparity for that pixel is invalidated and removed from the disparity map (set to zero). This is reasonable for the background and occluded areas, but for the foreground, it may cause holes. The threshold values are determined through manual tuning. The variance maps needed for the calculation are produced using three levels of recursion, similar to the matching process [8].

2.5. Achieving real-time performance

Optimization includes three major steps: compiler based optimization, system based optimization and assembly level optimization. Compiler based optimization performed to maximize the speed, exploits the architectural features such as pipelining, vectorization, and compiler intrinsic functions (e.g. `mult_fr1x16` and `add_fr1x16`).

System level optimization is achieved by partitioning the memory efficiently and streamlining the data flow. To achieve real-time performance, it is essential to consider the processing speed, data transfer rates and how the memory system handles the data during processing. There is a trade-off between the memory access speed and the available physical size of the memory array.

At 20 fps, for two gray scale images sub-sampled to QQVGA resolution, we would have $(160 \times 120 \text{ pixels/frame}) \times (1 \text{ byte/pixel}) \times (20 \text{ frames/sec}) \times (2 \text{ cameras}) \approx 750 \text{ KB/sec}$ of raw data throughput into the processor. This helps in estimating the amount of processing that is needed for the stereo vision system to function at 20fps. As mentioned in section 2.3, each processing core has access to only 64KB of fast L1 memory. Since each image

frame consists of 18.75KB of data, we can fit maximum of 3 images at the same time, within the L1 memory. To achieve a performance advantage, and leverage the dual-core architecture of the processor, we have implemented a two-staged pipelined programming model, where one core's output is the next core's input [8]. Processing task separation is optimized by Core A performing image acquisition, sub-sampling, stereo-rectification, and, Core B performing pre-processing, matching, and post-processing.

In a stereo-vision system, rectification and matching algorithms are the most time-consuming steps. Figure 6 illustrates how image sub-sampling and rectification are handled within Core A. The captured images are transferred into the L3 (external) memory using 2D DMA facility. Double buffering scheme allows Core A to process input images, while DMA transfers the next image frame into the external memory. This reduces the overhead on the processor. Using another 2D DMA operation, the image is sub-sampled and stored within the L1 memory.

$$(160 \times 120 \text{ pixels/frame}) \times (4 \text{ coefficients /pixel}) \times (2 \text{ bytes/coefficient}) = 150 \text{KB}$$

From the above, it can be seen, that the total amount of memory required for the two tables would be 375 KB. This far exceeds the memory space available on-chip, therefore, the lookup tables are stored in the external memory. We use double buffering along with parallel DMA transfers to solve the problem of long delays associated to external memory accesses (see Figures 3 and 6), resulting in more than 50% improvement in the execution time of the rectification algorithm. The last step involves transferring the final rectified image into the shared L2 memory, using 2D DMA.

In order to improve the performance of our matching algorithm (implemented on Core B), we have introduced an in-place processing approach to reduce the amount of memory required. In the in-place processing scheme, the source and destination memory locations remain the same. By keeping the processing core's access within the fast L1 memory, the associated performance bottlenecks and

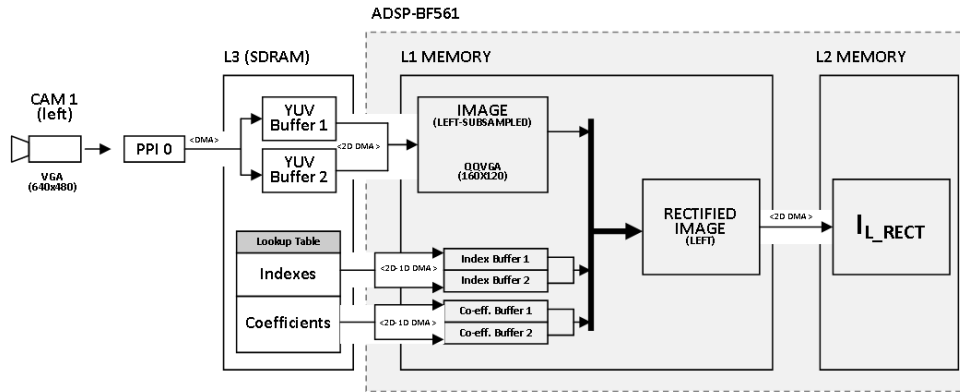


Figure 6. Memory and data flow management for Core-A.

During the rectification stage, constants like image indexes and coefficients have to be computed for the back-projection and bi-linear interpolation steps, respectively. These parameters need not be computed on the fly, as they take up a lot of core-clock cycles. Thus, the indexes and coefficients for each of the left and right images can be calculated and stored within the memory as a look-up table, to be used for rectification of the subsequent image pairs. The size of the index table can be computed as follows:

$$(160 \times 120 \text{ pixels/frame}) \times (2 \text{ index values/pixel}) \times (1 \text{ byte/index}) = 37.5 \text{KB}$$

Similarly, the size of the coefficients table can be computed as follows:

degradations caused by slow memory latencies, are alleviated.

We have also implemented the critical loops in assembly language, which allows us to leverage the efficient programming features provided by the BlackFin architecture such as specialized instructions (e.g. bit-counting facility for calculation of Hamming distance metric), utilization of multiple operations per cycle, hardware loop constructs, specialized addressing modes and interlocked instruction pipelines.

3. Power Consumption

In any portable, battery operated embedded system, special consideration has to be given to the management of power. Most of the stereo-vision algorithms are computationally intense and thus require the processor to run at the maximum core-clock frequency, in order to

achieve a faster frame rate. The algorithms have to be carefully selected and then tuned, in order to take advantage of dynamic power management. BlackFin processors have the ability to vary both the voltage, as well as the frequency dynamically, resulting in a significant reduction of power consumption. Dynamic power consumption is directly proportional to the square of the operating voltage and linearly proportional to the operating frequency, as shown in the equation 6 [17, 10] below:

$$P_{dyn} = KV_{DDINT}^2 \times f \quad (6)$$

where, P_{dyn} represents the dynamic power dissipation, K is a system constant, V_{DDINT} is the processor's core supply voltage, and f is the processor's clock frequency. Lowering frequency to reduce power consumption becomes beneficial only when the system is operating at the lowest voltage, as the code will take longer to execute at lower frequency values.

Our system incorporates features like intelligent voltage regulation, dynamic management of frequency, flexible power management modes (sleep, hibernate, full-on and active) and separate power domains for components. The usage of multiple power domains, one for the internal logic of the processor and the other for I/O, results in increased flexibility and power savings. External components were carefully selected, keeping their individual power dissipation in mind.

In addition, BlackFin VisualDSP++ tool suite comes with a Statistical Profiler that can quantify exactly how much time is spent in any given code segment. In order to improve performance, and reduce power consumption, the code blocks were carefully tuned. The average active power dissipation for the whole system can be calculated by summing the active power dissipated in individual power domains, and was found to be around 2.3W (700mA@3.3V).

4. Conclusion

The goal of this work was to improve upon the performance, quality and accuracy of MESVS-I. Through several optimization steps, we were able to achieve a real-time performance of 20fps. To enhance the robustness of the pre-processing stage to radiometric variations, we have employed census transform (size 3x3), which has been shown to out-perform rank (size 3x3 and 5x5). Based on the state-of-the-art embedded processor technology as its computational platform, we have developed a fully integrated and operational system which provides features such as power efficiency, compactness and low cost. Some of the applications include miniaturized mobile robotics, drowsy driver detection, and 3D object tracking etc.

5. Acknowledgements

The work is supported in part by the CRC program, the NSERC Discovery Grant, and the AUTO21 NCE.

References

- [1] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Belief Propagation for Early Vision", *International Journal of Computer Vision*, Vol. 70, No. 1, October 2006, pp. 41-54.
- [2] Y. Boykov, O. Veksler, and R. Zabih, "Fast Approximate Energy Minimization Via Graph Cuts", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 11, November 2001, pp. 1222-1239
- [3] C. Murphy, D. Lindquist, A. M. Rynning, T. Cecil, S. Leavitt, and M. L. Chang, "Low-Cost Stereo Vision on an FPGA", *IEEE FCCM*, April 2007, pp. 333-334
- [4] Y. Ruigang, M. Pollefeys, and L. Sifang, "Improved Real-Time Stereo on Commodity Graphics Hardware", *IEEE CVPR Workshop*, June 2004, pp. 36-44
- [5] L. D. Stefano, M. Marchionni, and S. Mattoccia, "A PCbased real-time stereo vision system", *International Journal of Machine Graphics and Vision*, Vol. 13, No. 3, January 2004, pp. 197-220
- [6] G. van der Wal, M. Hansen, M. Piacentino, "The Acadia vision processor", *Proceedings of 5th International Workshop on Computer Architecture for Machine Perception*, Padova, Italy, 2001, pp. 31-40
- [7] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister, "High-Quality Real-Time Stereo Using Adaptive Cost Aggregation and Dynamic Programming", *3DPVT* 2006, pp. 798-805
- [8] B. Khaleghi, S. Ahuja, and Q.M.J. Wu, "A New Miniaturized Embedded Stereo-Vision System (MESVS-I)", *CRV* 2008 (*to appear*).
- [9] "Selecting Processors for Video Applications", Berkley Design Technology Inc., available online at <http://www.bdti.com>
- [10] D. J. Katz, R. Gentile, "Embedded Media Processing", New York: Elsevier, 2006.
- [11] "Camera Calibration Toolbox for Matlab" available online at www.vision.caltech.edu/bouguetj/calib_doc/
- [12] R. Zabih and J. Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence", *ECCV* 2004, pp. 151-158
- [13] J. Banks, P. Corke, "Quantitative Evaluation of Matching Methods and Validity Measures for Stereo Vision", *IJRR*, 2001; 20; 512.
- [14] Cyganek, B., "Comparison of Nonparametric Transformations and Bit Vector Matching for Stereo Correlation", *Springer LNCS* 3322 (2004) pp. 534-547.
- [15] H. Hirschmuller and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching", *IEEE CVPR*, June 2007, pp. 1-8
- [16] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms", *International Journal of Computer Vision*, Vol. 47, No. 1-3, April 2002, pp. 7-42
- [17] "Estimating Power for ADSP-BF561 Blackfin@ Processors", Analog devices Engineer-to-Engineer note 293, available online at www.analog.com