

# A New Miniaturized Embedded Stereo-Vision System (MESVS-I)

Bahador Khaleghi, Siddhant Ahuja, and Q. M. Jonathan Wu  
*Intelligent Sensing Systems Laboratory, E.C.E. Department, University of Windsor*  
{khalegh, ahuja5, jwu}@uwindsor.ca

## Abstract

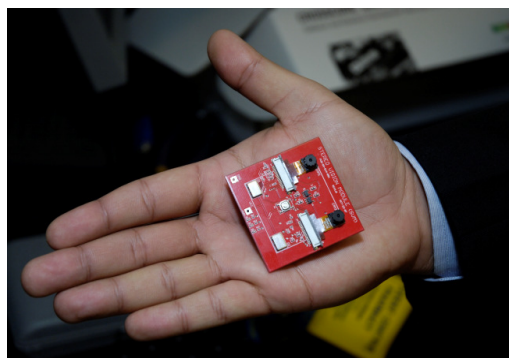
*We have developed a fully integrated, miniaturized embedded stereo vision system (MESVS-I) which fits into a tiny package of 5x5cm and consumes very low power (700mA@3.3V). The system consists of two small profile CMOS cameras, and a power efficient, dual-core embedded media processor, running at 600MHz per core. The stereo-matching engine performs sub-sampling, rectification, pre-processing using rank transform, correlation-based matching using three levels of recursion, L/R consistency check and post-processing. We have proposed a novel and efficient post-processing algorithm that removes outliers due to low-texture regions and depth-discontinuities by combining the contributions from the variance map of the rectified image, disparity map, and the variance map of the disparity map. To further enhance the performance of the system, we have implemented a two staged pipelined-processing scheme that takes advantage of the dual-core architecture of the embedded processor, thereby achieving a processing speed of around 10fps for disparity maps.*

## 1. Introduction

Stereo vision, i.e. the process of inferring 3D scene geometry from two or more images taken from different viewpoints, is an active and one of the most heavily investigated areas of computer vision. Stereo vision literature has significantly advanced, especially during the last decade, and many new algorithms have proven to be extremely useful in a variety of applications [1, 2]. Due to the complexity, cost, and/or power requirements of the hardware needed to implement them, the fundamental choice of the algorithm is restricted. There are numerous examples of real-time operational stereo vision systems in the literature. Most of these systems rely on FPGA [3, 4], PC [5, 6], Commodity Graphics [7] or a combination of those [8] as their computational platform. Some

researchers have also utilized the VLSI technology for developing application specific integrated circuit (ASIC) based systems [9]. Due to their high cost, lack of flexibility, and time-consuming and complicated design process, such systems are rarely implemented. A majority of aforementioned systems have focused on raw performance and not on the size and power demand, which are crucial in embedded settings and applications like miniaturized mobile robotics.

This paper presents the design and implementation of a new miniaturized and fully integrated embedded stereo vision system that was developed in our lab. The system is compact, power-efficient and of low-cost and uses state-of-the-art embedded media processor. We have miniaturized the system to fit within a tiny package of 5x5cm, and employed techniques to reduce the overall power consumption. Our system consumes around 700mA @ 3.3 V, and has near real-time performance of approximately 10 frames per second (see Figure 1).



**Figure 1. Prototype of Miniaturized Embedded Stereo Vision System (MESVS-I)**

The firmware incorporates: i) sub-sampling, ii) rectification, iii) a pre-processing stage based on rank transform, iv) correlation based matching with three levels of recursion, v) left/right consistency check to deal with image noise, radiometric variations, and half-occluded areas in the scene, and vi) a novel post-

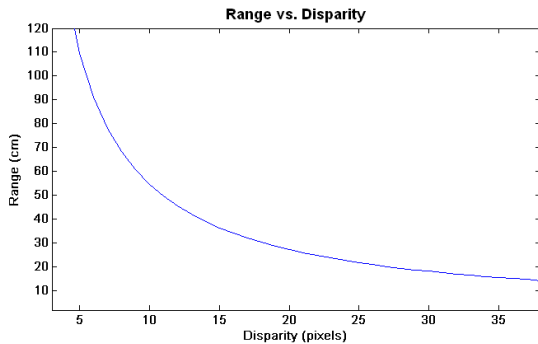
processing method to improve results around object boundaries and reduce number of mismatches caused by low texture regions, and depth discontinuities.

The paper is organized as follows. Section 2 provides a description of the system design, hardware and software implementation. The experimental results are presented in Section 3, which demonstrates the efficiency and robustness of our system. The conclusion is provided in Section 4.

## 2. System description

We have tried to keep the overall dimensions, power consumption and cost of the module as small as possible, while enhancing the accuracy and performance. One of the impacts of miniaturizing the system dimensions is on the baseline, i.e. the distance between camera's centers of projection. With the system dimension of 5x5 cm, the baseline of around 28mm was chosen. Due to the small focal length of the cameras, and the small baseline, we need to analyze the effect on the range of measurable objects in the scene (Horopter) and the corresponding accuracy (range resolution) to evaluate the feasibility of such configuration. The horopter must be large enough to encompass the range of objects in the application, and the accuracy associated with the obtained disparities within the specified horopter must be acceptable. The range uncertainty  $\Delta r$  is proportional to the range  $r$ , baseline  $b$ , focal length  $f$ , pixel size  $x$ , and the change in disparity  $\Delta N$ , as shown below:

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



where,  $N$  is the disparity value.

The result of our analysis is based on a baseline of 28mm, focal length of 2.8mm, and pixel size of 17um as shown in Figure 2a with horopter set to 5-35 and disparity range of 30, yielding a measurable range of about 15-100 cm. Figure 2b shows the range uncertainties vs. the range of interest (15-100 cm). For objects within close vicinity (less than 50 cm) the maximum uncertainty is around 5 cm. Our analysis shows that using a well adjusted horopter and a large disparity range, our system is capable of compensating for the impact of small baseline and is capable of retrieving range information with acceptable level of accuracy.

### 2.1 Hardware implementation

Embedded microprocessors may roughly be categorized into three groups: low-cost fixed-point, high-performance fixed-point and floating-point processors. Low-cost fixed-point DSPs operate at modest clock speeds (typically 350 MHz or less), and are mostly single-MAC devices that closely resemble the traditional DSP architectures of the early 1990's [10]. Floating-point DSPs such as Analog Devices SHARC<sup>®</sup> and TigerSHARC<sup>®</sup> families have complex circuitry and are not as energy-efficient as high-performance fixed-point processors, even though they offer high precision at a wide dynamic range.

High-performance fixed-point processors provide a fast yet power-efficient processor within an affordable price range. Most of them support multiple instruction executions using VLIW (very long instruction words) techniques, and also multiple MACs per cycle,

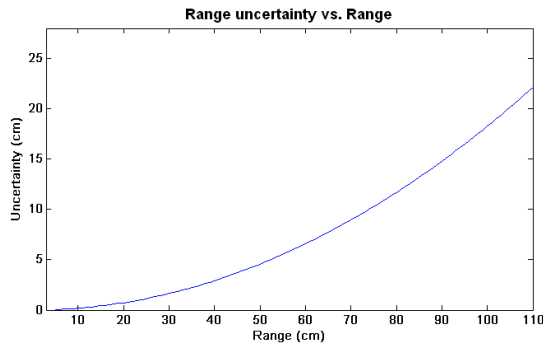


Figure 2. (a) Range vs. Disparity. (b) Range uncertainty vs. Range.

The range can be computed from the following equation:

$$r = \frac{bf}{Nx} \quad (2)$$

providing the best compromise amongst ease of programming, efficiency and performance at a lower cost.

As shown in Table 1, our candidate DSP from Analog Devices (ADI) BlackFin<sup>®</sup> family, ADSP-

BF561, is power efficient, compact and of lower cost, compared to the processor from Texas Instruments (TI) DaVinci<sup>®</sup> family. Although the Texas Instruments embedded processor offers higher computational power (based on higher MMACs), it costs as much as triple the cost for BlackFin<sup>®</sup>, requires more power and four times as much space. Considering the major requirements of our design, i.e. power efficiency, compactness and low cost, ADSP-BF561 has been chosen as the processing unit of our system.

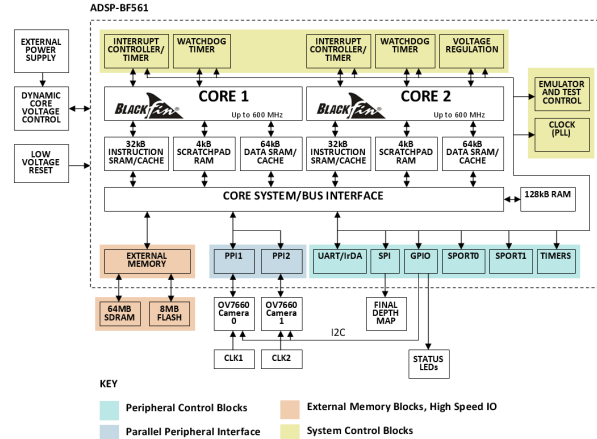
**Table 1. Comparison of candidate embedded media processors.**

DSP Model	BlackFin <sup>®</sup> ADSP-BF561	TMS320C6418- 600
Clock (MHz)	600x2	600
Power (mW)	1100 <sup>1</sup>	1850
RAM (Kbytes)	328	548
Camera interface	2 (ITU-656 compatible)	2 (ITU-656 compatible)
MMACs (Max)	2400	4800
Price (1k)	~17 USD	~55 USD
Dim. (W x L)	12 x 12 mm	23.1 x 23.1 mm

A high-level block diagram of our stereo vision system is presented in Figure 3. The system consists of ADSP-BF561 dual core processor, with each core capable of 1200 MMACs@600 MHz (2400 MMACs total). The ADSP-BF561 processor possesses a flexible parallel port interface (PPI), which is used to capture the images from two CMOS sensors in YUV 4:2:2 formats. Although color stereo matching outperforms matching of grayscale images, it would triple the memory requirements and increase the computational complexity [11]. In order to reduce the amount of memory required, and the associated computational burden, only the luminance channel *Y* data is used and the captured images are stored and processed as grayscale.

One of the other useful features of BlackFin<sup>®</sup> processor is its powerful direct memory access (DMA) capabilities that we have used extensively to avoid bottlenecks in video data movement within the system, and thus freeing the cores. The ADSP-BF561 also has 328 KB of on-chip (internal) memory and supports up to 4GB of external memory using 32-bit addresses. The internal memory is organized as 2x100 KB of L1 data and instruction memory dedicated to each core and accessible at full processor speed and 128 KB of L2 memory that is shared between the cores and slightly slower than L1 memory [12].

<sup>1</sup> An estimation assuming the core voltage of 1.2V,  $T_j = 100^\circ\text{C}$  and typical activity scenario [20].



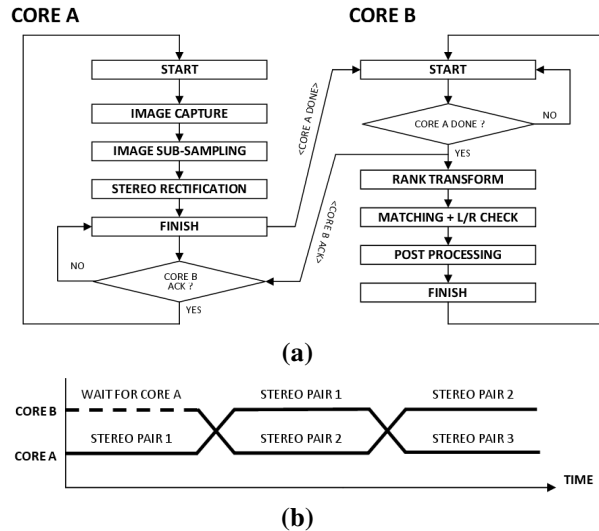
**Figure 3. Block diagram of MESSVS-I**

Additionally the system has 64MB SDRAM, 8MB of addressable flash memory, and dynamic core voltage regulator that allows adjusting core voltage via software. A JTAG interface is also incorporated to debug the system firmware. SPI port is used to transfer disparity map, 2D range data and high-level commands to other modules.

## 2.2 Pipelined dual-core processing

In order to take full advantage of our processor's dual core processing capabilities and reduce the processing time even further, the stereo matching engine is implemented as a two staged pipeline, as shown in Figure 4a. Core A of BlackFin<sup>®</sup> captures the images, sub-samples them, and then rectifies the images; while Core B handles the pre-processing, matching and post processing steps. To maintain synchronization between the cores, we use signaling through shared data protected by mutual locks available in BlackFin<sup>®</sup>.

The matching starts with Core A processing the first stereo pair and then signaling core B to begin performing the rest of the steps on the first stereo pair. Once started, Core B signals Core A back, allowing it to continue with processing the second stereo pair; while Core B finishes the processing of the first stereo pair and waits for the next signal from Core A. This procedure repeats for the subsequent stereo image pairs with Core A processing image pair  $n$  and Core B processing image pair  $n-1$  (see Figure 4b).



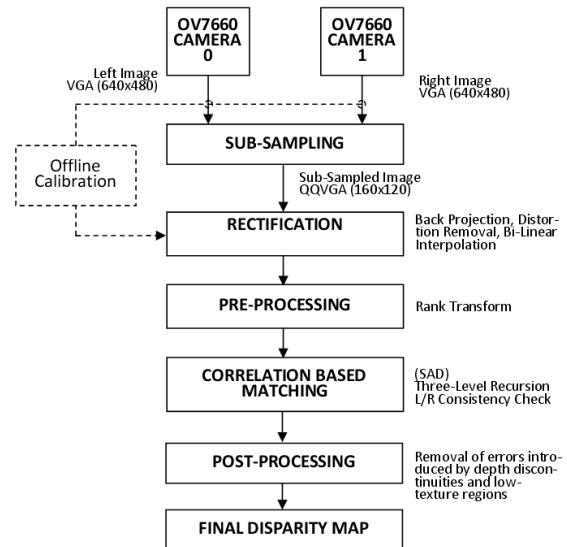
**Figure 4. (a) Two-staged pipeline processing flowchart (b) Core processing timeline for stereo image pairs.**

### 2.3 Stereo-Matching Engine

Stereo matching methods can be broadly categorized into local and global methods [13]. Global methods typically model disparity map constraints explicitly using energy function to be inferred (optimized) through 1-D (applied to image row as in scan-line optimization and dynamic programming) or 2-D (applied to the entire image as in belief propagation and graph cuts) optimization methods. They usually outperform local matching algorithms, however, due to their high computational complexity, they are not suitable for fast implementation except for methods based on dynamic programming [8].

Local methods can be subdivided into feature matching and block matching approaches. Feature matching methods operate by extracting viewpoint invariant features (e.g. edge, corner or line segments) and try to match them; while block matching methods typically calculate disparity map through local image correlation over a certain area (window). Feature matching methods can be implemented efficiently, but they do not provide dense disparity maps. Latter methods are capable of producing dense disparity maps and can be implemented effectively. Therefore, block matching methods are a good choice for real-time stereo vision systems involving applications that demand dense disparity maps. Our stereo system uses an efficient and robust correlation-based stereo matching engine to retrieve depth information of the scene in near real-time, i.e. 10 frames per second.

As illustrated in Figure 5, there are five major stages that constitute our software implementation, namely, image sub-sampling, stereo rectification, preprocessing, matching, and post-processing.



**Figure 5. Overview of the stereo matching engine.**

**2.3.1 Offline Camera Calibration.** We use the camera calibration toolbox for MATLAB, developed at the Institute of Robotics and Mechatronics at Caltech University [14]. The toolbox is user-friendly, powerful and well-documented. It supports a comprehensive camera model based on the pinhole model, provides estimates of uncertainties associated with parameters and is freely available to download for research purposes.

From our experiments, we found stereo calibration process to be a time-consuming stage, as the user has to manually select the four extreme corners of the calibration pattern. We also found that the placement of the planar calibration pattern, and the distortion model used, play a crucial role in the quality of the results obtained. For example, our experiments show that to have acceptable results, the radial distortion of cameras must be compensated. Adding tangential distortion to the model would only improve the results by a slight amount, yet considerably increase the computational complexity of the un-distortion process. Thus, in our implementation we only correct for radial distortion, in order to maintain a tradeoff between performance and quality of results.

**2.3.2 Sub-sampling.** Memory handling becomes a significant problem when dealing with image pairs on

an embedded media processor as the availability of fast, on-chip memory is limited. To deal with such a problem, external memory may be used to store an image pair; however the system performance will suffer due to high latencies. Thus, in order to maximize the performance, we have to ensure that the image data fits within the on-chip memory. The images were sub-sampled from VGA (640x480) resolution to QQVGA (160x120) resolution utilizing the 2-D DMA functions provided by the BlackFin<sup>®</sup> Processor's Internal Memory DMA (IMDMA) controller.

**2.3.3 Rectification.** In order to transform the image planes of left and right images, such that the pairs of conjugate epipolar lines become collinear and parallel to horizontal image axes, a rectification stage is necessary. This would help reduce 2-D correspondence search into simpler 1-D search along epipolar lines. Rectification involves back projection, image undistortion and bilinear interpolation. The main difficulty of implementing this stage is the need for floating-point operations, which is not supported by our fixed-point embedded media processor. Floating point operations can be emulated in software using the available library function calls (implemented in C), however, we found it to be inefficient.

To ensure the optimum performance of the rectification algorithm, we implemented a relaxed and fast floating-point routine in assembly language. Wherever possible, faster fractional data type operations (natively supported by BlackFin<sup>®</sup>) were used instead of floating-point operations, to further improve the efficiency of the algorithm. Based on our experimentation, such a strategy has resulted in over 70% reduction in the time taken to execute the rectification algorithm.

**2.3.4 Pre-processing.** To enhance the robustness of the matching algorithm, we have introduced a pre-processing stage that helps improve the quality of the final disparity map. The pre-processing stage is based on rank transform, which is a form of non-parametric local transform, invariant to changes due to image gain and bias. Rank transform  $R(P)$  of an image is defined as the number of pixels  $N(P)$  in the local region whose intensity  $I(P)$  is less than the intensity of the center pixel, as shown in the Equation 3 [15]. It replaces the intensity of a pixel ( $P$ ) with its rank among all the pixels within a square neighborhood around it.

$$R(P) = \|\{P' \in N(P) | I(P') < I(P)\}\| \quad (3)$$

Rank transform has been shown to outperform other methods including LOG filtering, normalized cross correlation (NCC), and mutual information (MI), on

images with simulated and real radiometric variations [16]. It can also be implemented efficiently on an embedded media processor.

**2.3.5 Matching.** Matching algorithm is carefully chosen, implemented, and optimized for performance, compactness, and robustness. It is based on a fast implementation of correlation-based matching that combines three levels of recursion to avoid redundant computations while speeding-up the matching process [17]. We deploy horizontal, vertical and modular recursions to efficiently compute correlation values, i.e. sum of absolute difference values (SAD) between correlation windows. We can calculate the SAD score,  $SAD(x,y,d)$ , between the left and right images using the equation below:

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

where, the window size is  $(2n+1) \times (2n+1)$  centered in the left image at  $(x,y)$  and in the right image at  $(x+d,y)$ .

SAD value for the subsequent rows  $SAD(x,y+1,d)$  can be calculated recursively using the current SAD value  $SAD(x,y,d)$  and an update term  $U(x,y+1,d)$  as shown in the Equation 5 [17].

$$SAD(x,y+1,d) = SAD(x,y,d) + U(x,y+1,d) \quad (5)$$

The update term  $U$  for vertical recursion is calculated by excluding the contributions of the row above the correlation window, denoted by  $y-n$ , and considering the contributions from the new row within the shifted correlation window, denoted by  $y+n+1$  as below [17]:

$$U(x,y+1,d) = \sum_{j=-n}^n |L(x+j,y+n+1) - R(x+d+j,y+n+1)| - \sum_{j=-n}^n |L(x+j,y-n) - R(x+d+j,y-n)| \quad (6)$$

Horizontal recursion can be computed in a similar way. Horizontal and vertical recursions can be combined using Equation 7 [17]. It can be seen that only the contributions associated with four pixels at the corner of the correlation window for both left  $L$ , and right  $R$  images, are needed to recursively compute the update term values, and their corresponding SAD values for subsequent rows.

$$U(x,y+1,d) = U(x-1,y+1,d) + (|L(x+n,y+n+1) - R(x+d+n,y+n+1)| - |L(x+n,y-n) - R(x+d+n,y-n)|) - (|L(x-n-1,y+n+1) - R(x+d-n-1,y+n+1)| + |L(x-n-1,y-n) - R(x+d-n-1,y-n)|) \quad (7)$$

From the above, we can conclude that the run-time of matching algorithm would be independent of window size, which in turn enhances the system flexibility. Another advantage of this approach is the

small amount of memory required, which is equal to  $WxD$  words, where  $W$  is the image width and  $D$  is the disparity range. We used the modular recursion as shown in Equations 8-10 [17]. It is based on the observation that contributions associated with the two pixels at the top right corner of correlation window is the same as the contributions of the top left pixels after  $2n+1$  iterations. Thus, the most recent  $2n+1$  terms can be temporarily stored within a 2D array  $M(x',d)$  for each disparity value  $d \in [0, d_{max}]$ , and can be re-used  $2n+1$  steps later to reduce the execution time of the algorithm [17].

$$\begin{aligned} U(x, y+1, d) &= U(x-1, y+1, d) + \\ &|L(x+n, y+n+1) - R(x+d+n, y+n+1)| - \\ &|L(x+n, y-n) - R(x+d+n, y-n)| - M(x', d) \end{aligned} \quad (8)$$

Array  $M(x',d)$  can be updated modularly as follows:

$$M(x', d) = |L(x-n-1, y+n+1) - R(x+d-n-1, y+n+1)| - |L(x-n-1, y-n) - R(x+d-n-1, y-n)| \quad (9)$$

where,

$$x' = x \bmod (2n+1), d \in [0, d_{max}] \quad (10)$$

For more details on the above, please refer to [17]. To treat the half-occluded regions in the scene, and enhance the robustness of the algorithm, we have incorporated an efficient implementation of left/right consistency check (LRC) method that produces accurate results under variety of conditions, compared to other occlusion detection methods available [18]. LRC check detects majority of the occluded pixels [17, 18] and performs quite well in highly textured scenes [18]. This is accomplished by temporarily storing the SAD values for each row within an array. In fact, we simply use the SAD values produced during the matching of right to left image, for cross-checking results. As a result, no additional memory is required to implement this validation step. For occluded regions, the disparity values are invalidated, and set to 0.

**2.3.6 Post-Processing.** Correlation-based matching methods have a tendency to blur object boundaries. This is a common issue also known as *foreground fattening problem* [19] that stems from the fact that the assumption of smooth disparity variations within correlation windows is typically violated around object boundaries. We propose a new post-processing approach designed with the following objectives:

- Alleviate errors due to depth discontinuities, which usually overlap with the object boundaries.

- Alleviate errors caused by poorly textured areas of the scene.
- Implement the post-processing algorithm efficiently, and with minimum overhead.

To satisfy the aforementioned objectives, our post-processing algorithm uses the recursively computed variance map of both the target image and the disparity map, and the original disparity map, to detect and remove outliers produced due to low-texture and depth discontinuous regions. The recursive procedure applied to compute variance maps is the same as one used for matching. Assuming a window size of  $N \times N$ , where  $N=2n+1$ , the mean value of the pixels  $\mu(x,y)$  in the intensity image  $I$ , is calculated using Equation 11:

$$\mu(x, y) = \frac{1}{N^2} \sum_{i,j=-n}^n I(x+j, y+i) = \frac{1}{N^2} S_1(x, y) \quad (11)$$

The variance  $\sigma^2(x,y)$  of the points within the window can be calculated by the following equation:

$$\sigma^2(x, y) = \frac{1}{N^2} \sum_{i,j=-n}^n (I(x+j, y+i) - \mu(x, y))^2 \quad (12)$$

For a symmetric window, it can be easily shown that the variance of the image can be calculated by simply summing up the square of intensity values and accumulating the original values for mean calculation, as shown in the equation below [17]:

$$\sigma^2 = \frac{1}{N^2} (\sum_{i,j=-n}^n I^2(x+j, y+i)) - \mu^2(x, y) = \frac{1}{N^2} S_2(x, y) - \mu^2(x, y) \quad (13)$$

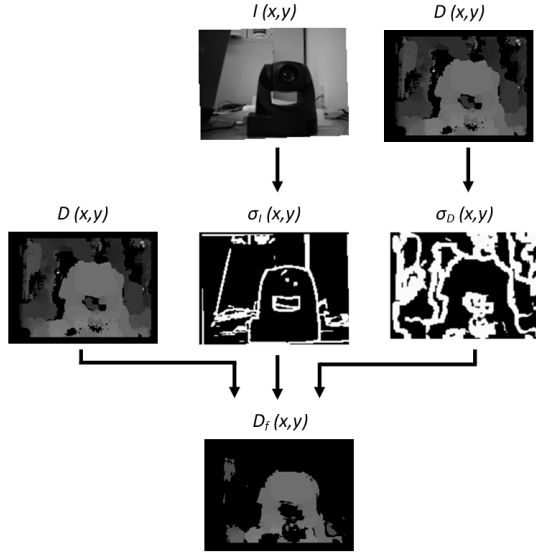
The summation terms  $S1$  and  $S2$  can be recursively calculated as follows [17]:

$$\begin{aligned} S_1(x, y+1) &= S_1(x, y) + U_1(x, y+1); \\ S_2(x, y+1) &= S_2(x, y) + U_2(x, y+1) \end{aligned} \quad (14)$$

The three levels of the recursion used to compute update terms  $U1$  and  $U2$  are similar to the ones already introduced for matching algorithm [17].

The idea behind the post-processing step is based on the observation that for pixels belonging to poorly textured regions of image, if the estimated disparity is not in harmony with other disparity values in its vicinity then it is most likely an erroneous match and must be removed. The low textured regions of image are shown in black, in the variance map  $\sigma_l(x,y)$  of the target image (see Figure 6).

Furthermore, the variance map of disparity map  $\sigma_D(x,y)$  is used as a measure of how well each disparity value agrees with the neighboring values, in a pre-defined window.



**Figure 6. Illustration of post-processing algorithm**

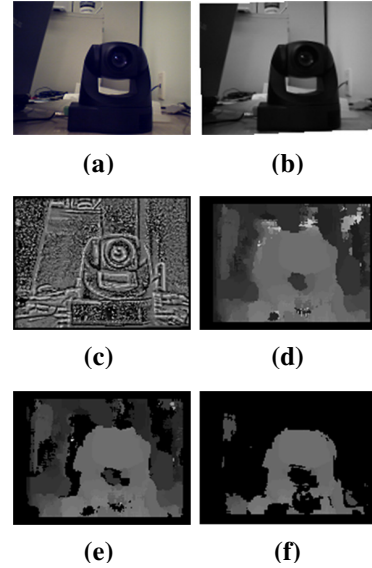
The estimated disparity value for a pixel  $D_f(x,y)$  is invalid and marked zero whenever its texture  $\sigma_I(x,y)$  is less than a threshold  $T_L$  and its disparity variation  $\sigma_D(x,y)$  is more than the threshold  $T_H$ . It will be marked as valid and will be retained otherwise. The values for  $T_L$  and  $T_H$  are manually tuned for best performance. Equation 15 shows the mathematical representation for the proposed idea.

$$D_f(x,y) = \begin{cases} 0, & \text{if } \begin{cases} \sigma_I(x,y) \leq T_L \\ \sigma_D(x,y) \geq T_H \end{cases} \\ D(x,y), & \text{otherwise} \end{cases} \quad (15)$$

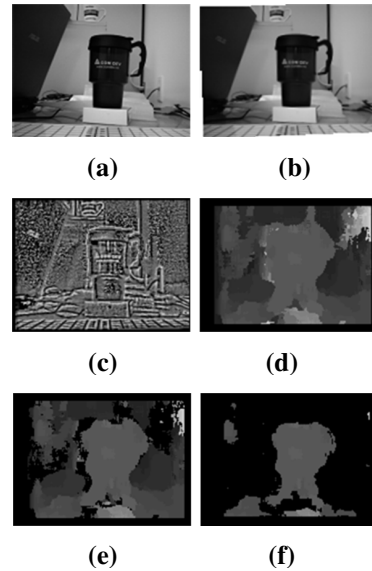
The proposed method improves the results obtained, specifically around object boundaries, as shown by the experimental results in the following section. Finally, disparities belonging to the background (more than 50 cm away) are removed by checking against a constant threshold.

### 3. Experimental Results

The results produced by our stereo matching engine are shown in Figure 7 and 8. The results presented in the figures are obtained with the window size of  $5 \times 5$  for rank transform, and  $15 \times 15$  for the correlation window. The disparity range is set to 30 with horopter of 5 to 35. Most of the erroneous matches associated with half-occluded regions of the scene (shown in black) are eliminated by LRC algorithm. Qualitative assessment of the post-processing step can be performed by observing the results in Figure 7f and Figure 8f. As can be seen, the shape of the object is clearly defined, especially around the curvature of the upper portion.



**Figure 7. Stereo matching engine output with camera as an object of interest. (a) Target Image, (b) Rectified Image, (c) Pre-processed image, (d) Initial disparity map, (e) Disparity map after L/R check, (f) Final disparity map after post-processing.**



**Figure 8. Stereo matching engine output with coffee mug as an object of interest. (a) Target Image, (b) Rectified Image, (c) Pre-processed image, (d) Initial disparity map, (e) Disparity map after L/R check, (f) Final disparity map after post-processing.**

It can be seen that our post-processing algorithm significantly alleviates the foreground fattening problem associated with correlation-based matching methods. Figure 7 shows the results obtained for a

scene with the camera as an object of interest, and Figure 8 is for coffee mug as an object of interest.

## 4. Conclusions

The experimental results presented in the previous section, show that MESVS-I is capable of producing satisfactory dense disparity maps. It can be deployed as a 2-D range measurement sensor in a miniaturized mobile robot to perform tasks such as obstacle avoidance, navigation and mapping.

This paper presented a fully integrated, miniaturized embedded stereo vision system (MESVS-I) which fits into a tiny package of 5x5cm and consumes very low power. The system is based on a power efficient, dual-core, pipelined, embedded media processor which performs sub-sampling, rectification, pre-processing, correlation-based matching using three levels of recursion and L/R consistency check. We have also shown that our new post-processing step effectively removes outliers caused by low-texture regions and depth-discontinuities.

## 5. Acknowledgements

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

## 6. 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] J. Yunde, Z. Xiaoxun, L. Mingxiang, and A. Luping, "A miniature stereo vision machine (MSVM-III) for dense disparity mapping", *ICPR*, August 2004, pp. 728- 731
- [4] 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
- [5] L. D. Stefano, M. Marchionni, and S. Mattoccia, "A PC-based real-time stereo vision system", *International Journal of Machine Graphics and Vision*, Vol. 13, No. 3, January 2004, pp. 197-220
- [6] H. Kim, D. B. Min, S. Choi, and K. Sohn, "Real-time disparity estimation using foreground segmentation for stereo sequences", *Optical Engineering*, Vol. 45, No. 3, March 2006, pp 037402 (10 pages)
- [7] Y. Ruigang, M. Pollefeys, and L. Sifang, "Improved Real-Time Stereo on Commodity Graphics Hardware", *IEEE CVPR Workshop*, June 2004, pp. 36-44
- [8] 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
- [9] 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
- [10] BDTI, "A Survey of Mainstream DSP Processors," April 2007; [www.dspdesignline.com/howto/198800216](http://www.dspdesignline.com/howto/198800216)
- [11] N. Chang, T.M. Lin, T.H. Tsai, Y.C. Tseng, and T.S. Chang, "Real-Time DSP Implementation on Local Stereo Matching", *IEEE ICME*, July 2007, pp. 2090-2093
- [12] "Blackfin Embedded Symmetric Multiprocessor ADSP-BF561 Datasheet" available online at [www.analog.com](http://www.analog.com)
- [13] 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
- [14] "Camera Calibration Toolbox for Matlab" available online at [www.vision.caltech.edu/bouguetj/calib\\_doc/](http://www.vision.caltech.edu/bouguetj/calib_doc/)
- [15] R. Zabih and J. Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence", *ECCV 2004*, pp. 151-158
- [16] H. Hirschmuller and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching", *IEEE CVPR*, June 2007, pp. 1-8
- [17] L. D. Stefano, M. Marchionni, and S. Mattoccia, "A fast area-based stereo matching algorithm", *Journal of Image and Vision Computing*, Vol. 22, No. 12, October 2004, pp. 983-1005
- [18] G. Egnal and R.P. Wildes, "Detecting Binocular Half-Occlusions: Empirical Comparisons of Five Approaches", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 8, August 2002, pp. 1127-1133
- [19] J. Lu, S. Rogmans, G. Lafruit, and F. Catthoor, "High-Speed Dense Stereo via Directional Center-Biased Windows on Graphics Hardware", *3DTV Conference*, May 2007, pp. 1-4
- [20] "Estimating Power for ADSP-BF561 Blackfin® Processors", Analog devices Engineer-to-Engineer note 293, available online at [www.analog.com](http://www.analog.com)