https://doi.org/10.2352/ISSN.2470-1173.2017.20.3DIPM-007

A comparison of stereo matching algorithms on multi-core Digital Signal Processor platform

Judicaël Menant
IETR - INSA of Rennes
jmenant@insa-rennes.fr

Jean-François Nezan
IETR - INSA of Rennes
jnezan@insa-rennes.fr

Luce Morin
IETR - INSA of Rennes
lmorin@insa-rennes.fr

Muriel Pressigout
IETR - INSA of Rennes
mpressig@insa-rennes.fr

Abstract

Stereo Matching algorithms reconstruct a depth map from a pair of stereoscopic images. Stereo Matching algorithms are computationally intensive. Implementing efficient stereo matching algorithms on embedded systems is very challenging. This paper compares implementation efficiency and output quality of the state of the art dense stereo matching algorithms on the same multicore embedded system. The three different classes of stereo matching algorithms are local methods, semi-global methods and global methods. This paper compares three algorithms of the literature with a good trade-off between complexity and accuracy : Bilateral Filtering Aggregation (BFA, Local Method), One Dimension Belief Propagation (BP-1D, Semi Global Methods) and Semi Global Matching (SGM, Semi Global Methods). For the same input data the BFA, BP-1D and SGM were fully optimized and parallelized on the C6678 platform and run at respectively 10.7 ms, 4.1 ms and 47.1 ms.

Introduction

Embedded vision is the merging of two technologies; embedded systems and computer vision. An embedded system is any microprocessor-based system that is not a general-purpose computer [1]. The goal of our work is to implement computer vision algorithms in modern embedded systems to provide them with real time stereo perception. Active devices such as Kinect [2] are able to produce real time disparity maps. This kind of device works by emitting an infra-red grid on the observed scene; the disparity map is deduced from this sensed-back grid. Those devices are limited to indoor use with a 5 meter range and they are sensitive to infra-red interferences. This paper focuses on Binocular Stereo-Vision algorithms to bypass these limitations. Stereo Matching aims to create 3D measurements from two 2D images, generating a disparity map which is inversely proportional to the distance of any object to the acquisition system. Disparity maps are used in scenarios where distance must be computed in all domains of computer vision. This is a strategically important knowledge field to test computation acceleration with.

Most existing real time implementations of stereo matching algorithms are carried out on desktop Graphical Processor Unit (GPU), leading to poor energy efficiency. On top of that GPU used for the implementations are different so that it is very difficult to truly compare the complexity of the algorithms. Embedded systems take up little space and consume little power, so they are ideal for widespread integration into everyday objects. Energy-efficient embedded platforms are now available. For instance the C6678 platform used in this article has a standard 10W power consumption. However, the architecture of embedded systems is significantly different to the architecture of desktop systems. The challenge is now to find and adapt algorithms and implementations that can fully exploit the powerful computational capabilities of such an architecture. The comparison of stereo matching algorithms exposed in this paper is made on the same embedded platform and it can be extrapolated to any other multi core embedded architecture with shared and distributed memory.

In this paper, three different stereo matching algorithms are considered. These three algorithms are at the state of the art in the field of stereo matching algorithms. Our work consists in adapting those algorithms to efficiently fit onto the C6678 platform. The implementations of Bilateral Filtering Aggregation (BFA) and One Dimension Belief Propagation (BP-1D) were previously studied in [3] and [4] respectively. The implementation of Semi Global Matching (SGM) is a new result in the field. The original SGM algorithm [5] uses a cost construction method based on "mutual information", the algorithm proposed in this paper uses cost construction based on Census which is more adapted to the targeted architecture. A contribution of the paper is also to fairly compare the implementation efficiency of the three algorithms in terms of quality and computing complexity using the same Digital Signal Processor (DSP) platform (C6678).

This paper is organized as follows : the principle of Binocular Stereo Matching algorithm and an overview of the C6678 are first introduced. The three algorithms and their optimization on the C6678 are then exposed. Results of the three algorithms are presented with respect to execution time and accuracy and finally perspectives and future works conclude the paper.

Stereo Matching principle

A Stereo Matching algorithm computes depth information from two cameras. The goal is to match a pixel in the left image with one in the right image. The disparity is the shifting of the two pixels in the left and the right images, the bigger the disparity, the closer the object from the two cameras. Depth information is thus retrieved from disparity.

There are two main classes of Stereo Matching algorithms, the dense ones and the parse ones. Sparse Stereo Matching algorithms consider only a set of interest points whereas dense stereo matching algorithms match all pixels. In this paper we consider dense Stereo Matching algorithms.

Dense Stereo Matching algorithms are mainly divided into three classes, local methods, global methods and semi-global methods [6]. Disparity computed with a local method depends only on colorimetric values of pixels within a finite window.

Global Stereo Matching algorithms maximize smoothness of the disparity map all over the image. Semi global methods maximize smoothness of the disparity map over a subset of image pixel that is not within a finite window. Global methods like Graph Cut come with good results in terms of accuracy but with a huge computational requirement incompatible with embedded systems and thus are not covered in this paper.

All dense Stereo Matching algorithms can be divided into three main parts :

- Cost construction measures the similarity of two pixels considering colorimetric values of those pixels and a finite neighbourhood.
- Cost aggregation takes into considerations other costs from the neighbourhood and other disparity values.
- Disparity selection decides what is the disparity value for a given pixel.

Three of the most promising Stereo Matching algorithms are studied in this paper:

- BFA is a fully local method.
- BP-1D is a semi-global method that considers each epipolar line separately (a single direction is considered).
- SGM is also a semi-global method that considers several directions in the right and left images.

The C66x multi-core DSP platform

The C6678 platform is composed of 8 C66x DSP cores and it is designed for image processing. The main features and constraints of this widespread multi-core platform have to be recalled to explain why the algorithms have to be modified to be efficiently executed on any embedded system.

Memory architecture

Memory is a critical point in embedded systems. Large memories being slower than smaller ones, modern systems integrate several memory layers in order to increase memory capacity without increasing access time. The memory hierarchy of the C6678 platform is composed of four memory layers :

- 512 MBytes of shared external DDR3 memory. This memory is slow, and the bus bandwidth is limited to 10 GBytes/s
- 4 MBytes of internal shared memory (MSMSRAM). It is a SRAM memory and it is faster (128 GBytes/s)
- 512 KBytes of L2 cache per core that can be configured as cache or local memory.
- 32 KBytes of (Program and Data) L1 cache per core.

SIMD

Single Instructions Multiple Data (SIMD) are instructions that are executed on multiple data. A SIMD instruction considers one or two registers (32 or 64 bits respectively) as a set of smaller words. For instance, an instruction that operates a 32-bit register as a group of 4 8-bit words is called a 4-way 8-bit SIMD instruction. This kind of instruction allows to manipulate several pixels (8-bits data) simultaneously.

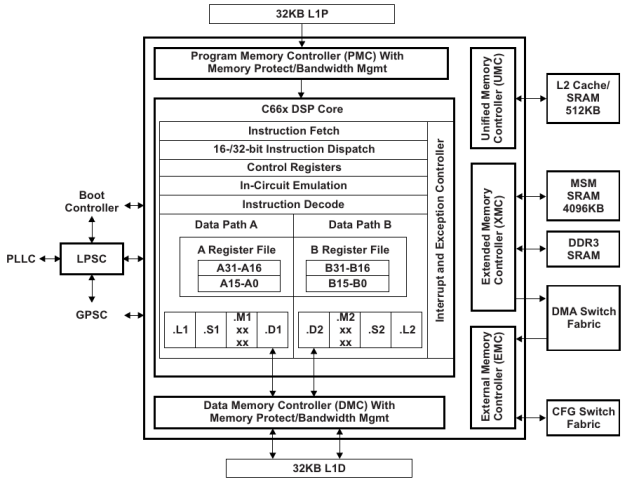


Figure 1: C66x DSP core architecture

Implementation strategies for stereo matching

All Stereo Matching algorithms have the same structure but different strategies can be applied to parallelized them. All Stereo Matching algorithms processing can be parallelized by diving images horizontally (slicing) without increasing the memory footprint. Nevertheless some algorithms can lead in synchronization penalties between slices.

Most of local Stereo Matching algorithms have their cost computed independently for each disparity level. Those algorithms can be parallelized along their disparity level. This parallelization is very effective but it comes with a higher memory footprint.

Specific DSP instructions can be used to increase implementation efficiency of algorithms : the bit count instruction for census, or the min instruction for saturation.

Finally, a pixel parallelization is possible to compute several pixels simultaneously thanks to SIMD instructions. Implementation and performances of this solution depend on the algorithm being used.

Next sections will describe the three considered stereo-matching algorithms which have been implemented and compared.

BFA stereo-matching algorithm

The first one of the considered stereo matching algorithms is close to the one proposed by Mei et al in [7] and [3]. The original algorithm is evaluated as one of the best performing local methods on the Middleburry[6] bench-marking data-base and has been modified for an efficient execution on multi-core embedded systems.

In the paper, this algorithm is denoted BFA for Bilateral Filtering Aggregation as its aggregation step is similar to bilateral filtering. This paper is based on an implementation of this algorithm on the C6678 platform provided by [8].

Cost construction

The cost construction step takes left and right images and computes a matching cost for all possible disparity levels. Its output is a cost map for each disparity level (same size as input image).

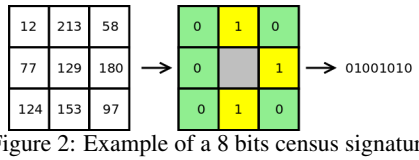


Figure 2: Example of a 8 bits census signature

The cost construction used in this algorithm is based on Census. Census produces an 8-bit signature for each pixel of an input image. As shown in figure 2, this signature is obtained by comparing each pixel to its 8 neighbours. Census signature is referred as cen_l, cen_r for respectively left and right grey level images.

For disparity d, pixel p of coordinates (x,y) is taken in the left image and compared to pixel p_d of coordinates $(x+d,y)$ in the right image. The result of the comparison is an error that is called matching cost. Equation (1) describes the matching cost that is computed on each pixel and for all possible disparities.

$$Cost(p,d) = \frac{1}{8}\sum_{k=0}^{7} \begin{cases} 0 & \text{if } cen_r(p)[k] = cen_l(p_d)[k] \\ 1 & \text{if } cen_r(p)[k] \neq cen_l(p_d)[k] \end{cases} \quad (1)$$

Where $Cost(p,d)$ is the cost associated to census, $cen_l(p)[k]$ and $cen_r(p)[k]$ are the k^{th} bits of the 8-bit census signature for pixel p in the left and right images respectively, p_d is the pixel of coordinates $p-d$, with d the disparity level.

The sum in Equation (1) is a sum of bit-to-bit "exclusive or" (XOR) boolean operation.

The output of the cost construction step is one cost map per disparity level. Those cost maps are the input of the cost aggregation step.

Cost aggregation

The cost construction step has a low computing cost, but it provides noisy matching cost maps. This noise is mainly produced by $Cost(p,d)$ which is random when compared regions are not correlated. To remove this noise, the cost aggregation step performs smoothing on areas with similar colour in the original image. The cost aggregation step is performed independently on each cost map. This is a key point regarding implementation.

The structure of the cost aggregation algorithm is similar to a bilateral filter. Cost aggregation is performed iteratively with varying parameters. It is defined by equation (2).

$$E_{d,i+1}(p) = \frac{W(p,p_+)E_{d,i}(p_+) + E_{d,i}(p) + W(p,p_-)E_{d,i}(p_-)}{W(p,p_+) + 1 + W(p,p_-)} \quad (2)$$

$E_{d,i}(p)$ is the cost map to be refined, $E_0(p)$ is the output of cost construction $Cost(p,d)$ in Equation (1).

Pixels p_+ and p_- have a position relative to pixel p :

- $p_+ = p+\Delta_i$
- $p_- = p-\Delta_i$

Equation (2) is computed alternatively for horizontal and vertical aggregations :

- When i is odd, the aggregation is vertical. The offset Δ_i is vertical.
- When i is even, aggregation is horizontal, the offset Δ_i is horizontal.

At each iteration the parameter Δ_i grows. Thus further pixels p_+ and p_- are used for smoothing p. Δ_i evolves according to equation (3), the influence range is limited by the modulo (here $\Delta_i \in [0,32]$) [3].

$$\Delta_i = floor(i/2)^2 \bmod 33 \quad (3)$$

Weights W in equation (2) are defined by equation (4) :

$$W(p_1,p_2) = e^{C_d.\Delta_i - \frac{\sqrt{\Sigma_{col\in(r,g,b)}(I_r col(p_1)-I_r col(p_2))^2}}{L_2}} \quad (4)$$

Where C_d is a weight applied to distance [7] and L_2 is the weight applied to similarity [7]. $I_r\{r,g,b\}$ and $I_l\{r,g,b\}$ are the RGB (Red, Green, Blue) signals of right and left images.

Disparity selection

The disparity selection step minimizes the matching cost. To do so, the Winner Takes All (WTA) strategy [6] is applied. The WTA strategy is a simple arithmetic comparison expressed by :

$$Disp(p) = \underset{d\in[0,Ndisp]}{\text{argmin}} \; E_{d,Nit}(p) \quad (5)$$

Equation 5 expresses the fact that the smallest cost gives the best disparity choice, that is why the *argmin* operator is used.

The output of disparity selection is a dense disparity map providing an integer disparity value for each pixel in the right image.

Implementation

The census cost is implemented using specific DSP instructions XOR and bitcount (count the number of bits equal to 1 in a word) that comes in SIMD version. This allows to compute a pixel cost from two census signatures in 0.25 machine cycles.

The weights $W(p1,p2)$ are precomputed, they only depend on the input image. The bottleneck of BFA is the memory bandwidth. The fact that $W(p+,p) = W(p,p-)$ allows to reduce the size of those precomputed buffers by a factor 2. They then fit into the MSMSRAM, reducing the memory bottleneck.

BP1D stereo-matching algorithm

The BP-1D algorithm is a semi global Stereo Matching algorithm. Cost construction is based on a Sum of Absolute Difference (SAD), it has no disparity selection. Indeed, its disparity selection is based on a belief propagation to resolve an Homogeneous Markov Chain (HMC) and thus it takes into account the neighbourhood. Each line is considered independently. This allows the reduction of memory complexity of the algorithm and enhances parallelism.

Cost construction

The cost $C_{sad}(p,d)$ is defined as the **SAD!** (**SAD!**) of two pixels p and $p-d$ of respective coordinates (x,y) and $(x-d,y)$:

$$C_{sad}(p,d) = \sum_{col\in(r,g,b)} \|I_r col(p) - I_l col(p-d)\| \quad (6)$$

Disparity selection

BP-1D disparity selection is based on the optimization of a HMC [9, 10] executed line by line. BP-1D has also another specificity : it works on the Cyclopean view. A Cyclopean view is the view that would be obtained by an hypothetical camera that is exactly between left and right cameras. Figure 3 explains the principle of the Cyclopean view. This view is very convenient : the position and the disparity can be obtained very easily from this view and all possible neighbours (in space or disparity) are near each other. Moreover, occultations are easily deduced from this view.

The BP-1D algorithm manages occulted areas : each disparity for each pixel has three possible states with their corresponding cost, *ML*, *MR* and *B* for respectively a pixel only on the left image, a pixel only on the right image and a pixel on both images.

Each node $u_x(d,s)$ of the Graph of Profile Variants (GPV) is related to one pixel in the left image $I_lrgb(x_1)$ and one pixel in the right image $I_rrgb(x_2)$ such that $x = (x_1 + x_2)/2$ and $d = x_1 - x2$. Each pixel x for each possible disparity level $d \in [0, D-1]$ has 3 different possible hidden states $s \in \{ML, MR, B\}$.

The line in Figure 3 is an example of a Profile Variant. A Profile Variant is a line that associates a disparity d for each value of x. A GPV is a set of all possible Profile Variant lines. Due to ordering and visibility constraints (occulations), the admissible transitions in the nodes $u_x(d,s)$ of the Graph of Profile Variants are limited.

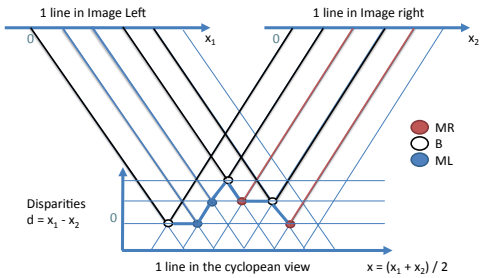


Figure 3: Example of energy minimisation in a cyclopean view

BP-1D aims at finding the best Profile Variant in the GPV, i.e the Profile Variant minimizing an energy function E^*. For a well-posed energy minimization problem BP-1D gives the unique optimizer **u***. As the Stereo Matching is an ill-posed problem, BP-1D gives one of the several possible solutions.

The energy is the sum of several costs enabling to take into account the neighbourhood of the node in the HMC : F is the forward message, ϕ the backward message and $C_{sad}(p,d)$ a cost associated to each node.

$$E^* = \sum_{x=1}^{n} \min_{d,s} E(u_x(d,s)) \tag{7}$$

$$E(u_x(d,s)) = F_x(d,s) + \phi_x(d,s) + C_{sad}(x,y,d) \tag{8}$$

Taking into account the possible transitions, the forward message $F_x(d,s)$ is computed in a forward pass using the following equations :

$$F_x(d,ML) = F_x(d,B) = \min \begin{cases} F_{x-0.5}(d-1,ML)) + \varphi_0 \\ F_{x-1}(d,B)) + C_{sad}(x-1,y,d) \\ F_{x-1}(d,MR)) + \varphi_0 \end{cases} \tag{9}$$

$$F_x(d,MR) = \min \begin{cases} F_{x-0.5}(d+1,B)) + C_{sad}(x-0.5,y,d+1) \\ F_{x-0.5}(d+1,MR)) + \varphi_0 \end{cases} \tag{10}$$

The backward message $\phi_x(d,s)$ is computed in a backward pass using the following equations :

$$\phi_x(d,ML) = \min \begin{cases} \phi_{x+0.5}(d+1,ML)) + \varphi_0; \\ \phi_{x+0.5}(d+1,B)) + C_{sad}(x+0.5,y,d+1) \end{cases} \tag{11}$$

$$\phi_x(d,B) = \phi_x(d,MR) = \min \begin{cases} \phi_{x+1}(d,ML)) + \varphi_0 \\ \phi_{x+1}(d,B)) + C_{sad}(x+1,y,d) \\ \phi_{x+0.5}(d-1,MR)) + \varphi_0 \end{cases} \tag{12}$$

For our experiments, the φ_0 cost has been fixed to 40. φ_0 is linked to the dynamic of input signal. This constant defines the smoothing constraint it gives a penalty for each change in disparity levels.

The disparity selected for each node $u_x(d,s)$ is the one minimizing the energy $E(u_x(d,s))$

$$D(x) = \underset{d \in [0,N-1[}{\operatorname{argmin}} E(u_x(d,s)) \tag{13}$$

Implementation

The BP-1D algorithm can be parallelized easily without increasing memory consumption because it considers each line independently. This is the key point of this algorithm regarding its implementation. BP-1D has been parallelized without any particular framework, because it does not need any special synchronisation (each line is treated independently).

The disparity selection part uses additions and minimums. The C66x core provides efficient SIMD instructions to implement those costs.

SGM stereo-matching algorithm

The SGM algorithm [5] is one of the best non-global methods [6]. It gives a good trade-off between output quality and execution speed.

Cost construction

The original SGM cost construction is based on mutual information [5]. A second version of this algorithms uses a Census based cost construction [11]. The census based version is preferred in this paper for performance reasons : mutual information needs an initial disparity map as input and it is iterative. Moreover, the census implementation, as explained in the BFA section, fits very efficiently on the C6678 architecture.

The census cost used in this implementation is described in Equation (1).

Disparity selection

SGM disparity selection is based on an Hidden Markov Model (HMM) like BP-1D. But instead of aggregating horizontally only, the aggregation is performed in several directions. The BP-1D algorithm produces horizontal striding artefacts in its output (as shown in figure 4f). By considering several direction paths instead of one, SGM avoids this artefact.

Table 1: Dynamic of the SGM cost matrix

Image	Teddy	Cone	Sawtooth	Art
Costs sup. 255	0.2%	0%	0.3%	2.7%

Equation (14) gives the aggregation cost for one direction r. P_1 and P_2 set the desired of disparity map smoothness, they are penalties for small and large discontinuities and are set in this paper at respectively 20 and 40. $L_r(p,d)$ is the cost aggregated along direction r for disparity d at pixel p. $C(p,d)$ is the cost at pixel p and disparity d. $p-r$ is the previous pixel before p along direction r.

$$L_r(p,d) = C(p,d) + \min\Big(L_r(p-r,d), L_r(p-r,d-1)+P_1, L_r(p-r,d+1)+P_1, \min_i(L_r(p-r,i))+P_2\Big) - \min_k(L_r(p-r,k)) \quad (14)$$

To obtained the disparity map, the energy on all direction r is minimized :

$$Disp(p) = \underset{d\in[0,Ndisp]}{\operatorname{argmin}} \sum_r L_r(p,d) \quad (15)$$

Implementation

By subtracting $\min_k(L_r(p-r,k))$ from the cost propagation along one path, Equation (14) has a bounded dynamic [5]. The original paper [5] proposed an implementation that uses 16 bits words. An 8-bit words implementation coupled with saturated arithmetic is proposed in this paper. Because costs are minimized, only the smallest values are selected. The 8-bit saturated arithmetic only impacts largest values and thus has a negligible impact on quality. Moreover as shown in table 1, as most of the values fit on 8 bits, the output quality is not degraded by this modification.

The implementation proposed in this paper stores all costs in 8-bit words and uses saturated arithmetic to avoid overflows. By doing this the memory constraint is divided by two and memory transfers are boosted. The use of SIMD instructions is also much more efficient. Indeed the C6678 SIMD allows to handle up to 8 8-bit words in one instruction.

In Equation (14), each series of costs along one direction is independent. For instance, when the direction is horizontal, each line of costs is treated independently. Thus, in the proposed implementation, each core of the C6678 works on one series of costs along one direction, and then only need a subset of the cost matrix that fits in its local L2 memory.

Results

This section presents results in quality and execution time of the previously introduced stereo matching algorithms.

Table 2 expresses the quality of the three different algorithms and their respective execution speed on the 8 cores of the C66x platform. The quality is expressed in percentage of bad pixels rated by the Middlebury evaluation algorithm [6]. The percentage of bad pixels is an average on 6 Middlebury images (*cones*, *map*, *sawtooth*, *teddy*, *tsukuba* and *venus*). Execution time is given for the *sawtooth* image and not an average on all images because the execution time is predictible reagrding the size of its input. That means, knowing the execution time for a size of image and a number of disaprity, all execution time can be found (the complexity is $X \times Y \times NDISP$).

Table 2: Algorithms speed and output quality (the lowest the better)

Algorithm	Speed (FPS)	Quality (% of bad pixels)
BFA	93.5	4.02 %
BP-1D	245	3.81 %
SGM	21.2	8.60%

Table 3: Algorithms profiling

Algorithm	Construction	Aggregation	Selection
BFA	24 %	72 %	5 %
BP-1D	3 %	97 %	
SGM	13 %	70 %	17 %

Table 3 expresses the relative execution time of each main part of each algorithm. These results are obtained with one core of the C66x platform clocked at 1GHz.

The parallelization efficiency of each algorithm on the C66x platform is given by table 4. All execution time are for the C66x platform.

The BP-1D algorithm gives the best quality and execution time results (see table 2). This can be explained by the fact that it can be easily parallelized (table 4). Each line is processed separately in BP-1D. That is why there is a speed up factor (table 4) that is equal to the number of cores. Each line is stored in the local memory cache, and it results in a better cache use. The main drawback of BP-1D is the fact that it cannot be tuned to match particular time or quality requirements. But BP-1D fits very well on a multi-core architecture.

Compared to the original implementation, output quality is good. BFA execution time on a desktop GPU (Nvidia Quadro FX 3700) is 192 ms [3] for a 450 x 375 pixel image with 59 disparity levels. The extrapolated result for *sawtooth* is 64 ms. The DSP version is 6 times faster than its GPU version.

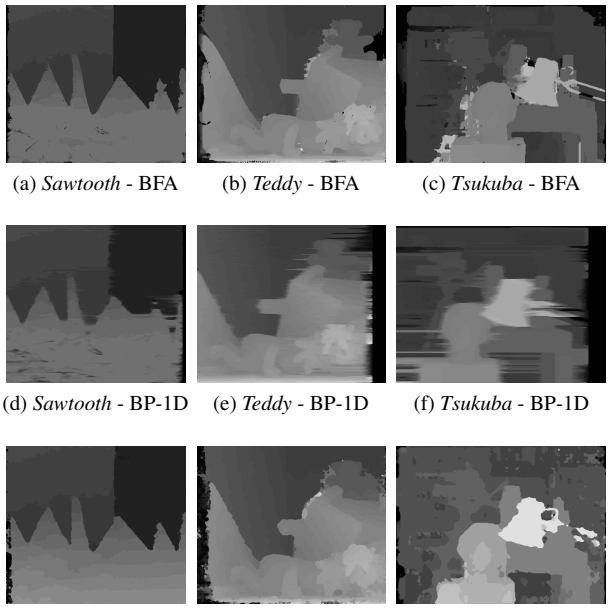
SGM is executed at 4.5 Hz [11] onto a desktop GPU (NVidia GeForce GTX 275) for a 640 x 480 pixel image and 128 disparity levels. The extrapolated result for *sawtooth* is 17 ms. The GPU version is faster 3 times than the DSP version, but the DSP consumes 25 times less power than the GPU.

BFA and SGM algorithms are parallelized thanks to Open Multi-Processing (OpenMP), whereas BP-1D is parallelized directly on the DSP, without any framework. This is why parallelized results are better for BP-1D. BFA and SGM algorithms are less data independent than BP-1D and thus require a framework to handle parallelization (OpenMP).

Table 3 shows the specificity of each algorithm. For BP-1D, the most computing intensive part is cost aggregation which cannot be separated from disparity selection. It is the most computing intensive part. For SGM and BFA, there is a trade-off between cost construction and cost aggregation. This gives more possibilities for further improvement on those algorithms.

Table 4: Multicore Execution Time (ms)

Algorithm	1 core	2 cores	4 cores	8 cores
BFA	52 (x1)	29 (x1.8)	16 (x3.2)	11 (x4.9)
BP-1D	41 (x1)	20 (x2)	10 (x4)	5 (x8)
SGM	330 (x1)	174 (x1.9)	92 (x3.6)	47 (x7)



(a) *Sawtooth* - BFA (b) *Teddy* - BFA (c) *Tsukuba* - BFA

(d) *Sawtooth* - BP-1D (e) *Teddy* - BP-1D (f) *Tsukuba* - BP-1D

(g) *Sawtooth* - SGM (h) *Teddy* - SGM (i) *Tsukuba* - SGM

Figure 4: Output disparity maps

Figure 4 provides output disparity maps for each algorithm considered in this paper. Each image has been post-processed with the following post treatment :

- A 3 by 3 median filter for BFA and SGM.
- A median filter for BP-1D which is larger in the vertical direction (3 by 11) to reduce the strides on disparity maps.

BFA gives good visual results but poorly manages occulted areas. The BP-1D gives better results according to Middlebury criteria because of its ability to handle occulted areas but its visual quality is decreased by its striding effect. Output disparity maps of BP-1D have strides because each line is processed independently, and there is no coherency between lines.

Conclusion

Stereo Matching algorithms are very resources intensive. The goal of this paper is to compare the efficiency of different Stereo Matching algorithms on a DSP platform. The C6678 platform is one of the most powerful DSP platforms on the market. That is why this platform has been chosen for this comparison.

Three different kinds of algorithms have been implemented on this platform. BFA, a local algorithm, and two semi-global algorithms : BP-1D and SGM.

All three different algorithms parallelize well on the C6678 platform. BP-1D uses more efficiently the 8 cores with a speed-up of 8 because each line is processed independently. BFA and SGM require inter-core synchronization that reduces efficiency of multi-core implementation with respectively a speed-up of 4.93 and 7.02.

Finally, for the same input image the BFA, BP-1D and SGM with an optimized implementation on the C6678 platform run at respectively 10.7 ms, 4.1 ms and 47.1 ms. The BP-1D algorithm provides the best results in quality and execution speed. But BFA and SGM can be tuned more easily by modifying their parameters such as number of iterations or number of paths.

Several improvement can be made on those algorithms. All algorithms have parameters that have to be tuned. In this paper, these parameters are set according to the literature. Those parameters may be optimized to have better quality results without any performance lost.

References

[1] E. V. Alliance, "What is embedded vision." http://www.embedded-vision.com/.

[2] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.

[3] J. Zhang, J. F. Nezan, M. Pelcat, and J. G. Cousin, "Real-time gpu-based local stereo matching method," in *Design and Architectures for Signal and Image Processing (DASIP), 2013 Conference on*, pp. 209–214, Oct 2013.

[4] J.-F. Nezan, A. Mercat, P. Delmas, and G. Gimelfarb, "Optimized belief propagation algorithm onto embedded multi and many-core systems for stereo-matching," *Processdings of the 24th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, 2016.

[5] I. Ernst and H. Hirschmüller, "Mutual information based semi-global stereo matching on the gpu," in *Proceedings of the 4th International Symposium on Advances in Visual Computing*, ISVC 08, (Berlin, Heidelberg), pp. 228–239, Springer-Verlag, 2008.

[6] R. S. Daniel Scharstein, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, no. 47, pp. 7–42, 2002.

[7] X. Mei, X. Sun, and M. Zhou, "On building an accurate stereo matching system on graphics hardware," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pp. 467 –474, nov. 2011.

[8] J. Menant, M. Pressigout, L. Morin, and J.-F. Nezan, "Optimized fixed point implementation of a local stereo matching algorithm onto c66x dsp," in *Design and Architectures for Signal and Image Processing (DASIP), 2014 Conference on*, pp. 1–6, Oct 2014.

[9] G. Gimel'farb, "Intensity-based computer binocular stereo vision: Signal models and algorithms," *International Journal of Imaging Systems and Technology*, no. 47, pp. 189 – 200, 1991.

[10] J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo matching using belief propagation," vol. 25, pp. 787 – 800, july 2003.

[11] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, pp. 1582–1599, Sept 2009.

Author Biography

Judicael Menant graduated from INSA of Rennes in 2012. Then he worked for Renesas, a company that designs System On Chip for smart phones. Judicael Menant Started his PhD at IETR on January 2014. His work has focused on Stereo Matching algorithm onto embedded systems.