Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2007, Article ID 81387, 8 pages
doi:10.1155/2007/81387

Research Article

Real-Time 3D Face Acquisition Using Reconfigurable Hybrid Architecture

Johel Mitéran, Jean-Philippe Zimmer, Michel Paindavoine, and Julien Dubois

Le2i Laboratory, University of Burgundy, BP 47870, 21078 DIJON Cedex, France

Received 2 May 2006; Revised 22 November 2006; Accepted 12 December 2006

Recommended by Joern Ostermann

Acquiring 3D data of human face is a general problem which can be applied in face recognition, virtual reality, and many other applications. It can be solved using stereovision. This technique consists in acquiring data in three dimensions from two cameras. The aim is to implement an algorithmic chain which makes it possible to obtain a three-dimensional space from two two-dimensional spaces: two images coming from the two cameras. Several implementations have already been considered. We propose a new simple real-time implementation based on a hybrid architecture (FPGA-DSP), allowing to consider an embedded and reconfigurable processing. Then we show our method which provides depth map of face, dense and reliable, and which can be implemented on an embedded architecture. A various architecture study led us to a judicious choice allowing to obtain the desired result. The real-time data processing is implemented in an embedded architecture. We obtain a dense face disparity map, precise enough for considered applications (multimedia, virtual worlds, biometrics) and using a reliable method.

1. INTRODUCTION

We present in this paper a comparison of numerous methods allowing obtaining a dense depth map of human face, and the real-time implementation of the chosen method. Acquiring 3D data of human face is a general problem which can be applied in face recognition [1–3]. In this particular case, the knowledge of depth map can be used for example as a classification feature. It can be seen as an improvement of classical method such as eigenfaces [4]. The stereovision technique we used is well known and consists in acquiring data in three dimensions from two cameras. The key problem in stereo is how to find the corresponding points in the left and in the right image [5] (correspondence problem). Many research activities are currently dealing with stereovision, using different approaches to solve the correspondence problem. Since our main application is face recognition, we studied different methods adapted to this problem. Moreover, our applications have to be completed in real-time (10 image/s). General purpose computers are not fast enough to meet these requirements because of the algorithmic complexity of stereovision techniques. We studied the implementation using hybrid approach. Although various implementations have already been considered [6, 7], we propose a simple real-time implementation, including a regularization step, based on a multiprocessor approach (FPGA-DSP) allowing to consider an embedded and reconfigurable processing. Faugeras et al. [6] proposed a multi-FPGA (23 Xilinx 3090) architecture which is too complex for an embedded application. Ohm and Izquierdo [7] proposed a stereo algorithm where dense map is obtained using bilinear interpolation from global disparity estimation. However, this approach used for face localization is not enough precise for face recognition problem. In [8], Porr et al. used the Gabor-based method implemented in a software and hardware system. The board is virtex-based as ours, but does not allow embedded post processing as we do in the DSP from Texas Instrument. We present in the first part of the paper the study of the whole necessary processing, while reviewing and comparing various employed methods. In the second part, we present the implementation on an embedded architecture of our method which provides depth map of face, dense and reliable.

2. METHOD

2.1. Stereodata processing flow

The main goal of this whole processing is to match corresponding points between two images. The distance or disparity between these homologous points is then calculated.

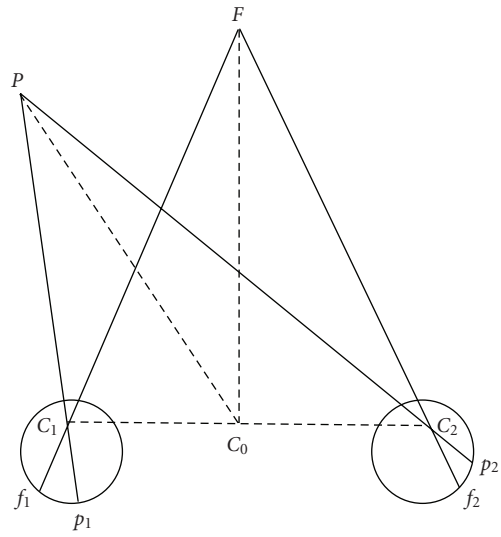


Figure 1: Retina disparity.

This value is proportional to the depth, thus codes the third dimension (Figure 1).

The retina disparity D is defined as follows:

$$D = E(f_1, p_1) - E(f_2, p_2), \tag{1}$$

where $E(x, y)$ is the Euclidian distance between x and y.

This value is proportional to the depth difference between P and F.

The processing flow is composed of two main parts. The first requires mainly geometrical criteria and modeling, the second uses signal processing knowledge.

The first part is the camera calibration, either for each one in 3D space (strong calibration) or relatively between them (weak calibration and epipolar geometry). To this stage can be added a rectification image processing. This rectification allows to match the image lines of the stereo pair, and thus to work in only one dimension [5].

The second part consists in the homologous points matching. Various methods were developed to constitute dense depth maps. Two papers [9, 10] present a large review of these techniques. Since the goal of this paper is mainly to present the hardware implementation of our solution, we will only recall the principle of the 3 methods we compared, the results of this comparison which will justify our final choice.

2.2. Principal methods of dense depth maps constitution

Several methods have been studied and give interesting results. We can classify them in three principal parts: the methods based on partial differential equation (PDE) [11], on local phase [12], and on crosscorrelation [13].

2.2.1. Partial differential equations

This method is based on the minimization of an energy criterion by solving a diffusion equation. Various implementations were given. One of them provides the depth by resolution of the discrete Euler-Lagrange equation [14]. A judicious choice of the regularization function allows preserving discontinuities [11]. A multiresolution result is obtained by iteratively searching for the solution. In order to obtain efficient solutions, it can be here interesting to introduce the epipolar constraint.

The methods based on PDE allow obtaining dense depth maps and a very good precision on the results. Unfortunately, these processing require too significant computing times and cannot, yet, be considered on a simple embedded architecture. Therefore, we did not include this method in our comparison.

2.2.2. Crosscorrelation

This classical method is based on homologous points matching by search of the minimum of a criterion by crosscorrelation in shifting local windows [13]. The most usual criteria used are the crosscorrelation or the square difference (or the difference absolute value) of the pixel intensities between each image of the stereopair. This method can be improved, in order to make it less sensitive to the differences between the average gray level of the two images, by centering and/or by a local normalization. Moreover, the criterion is applied in a local window surrounding the tested pixels. The criterion $C_{x,y}$ is then computed as follows:

$$C_{x,y} = \sum_{-l\le i\le l;\, -h\le j\le h} \left(\left(I_1(x+i, y+j) - \overline{I}_1(x, y) \right) - \left(I_2(x+i, y+j) - \overline{I}_2(x, y) \right) \right)^2, \tag{2}$$

where $I_1(x, y)$ is the pixel luminance of left image, $I_2(x, y)$ is the pixel luminance of right image, and h and l are, respectively, the height and length of the local window centred in (x, y). $\overline{I}_1(x, y)$ and $\overline{I}_2(x, y)$ are the mean of luminance computed in these local windows.

The method of shifting window processing requires a range of limited disparities $[d_1, d_2]$. The criterion is then calculated for each disparity. The maximum criterion gives the required disparity. If the maximum is obtained for d_1 or d_2, an error value is affected to D (Figure 2).

This processing is carried out effectively in one dimension and thus requires either to know the epipolar constraint, or to work on rectified images. A double processing Left Image/Right Image then Right Image/Left Image, followed by a validation step, makes it possible to remove wrong matching.

A multiscale approach can also be considered, allowing an extension of the range of the required disparities and a validation at various scales in order to obtain better results on poorly textured patterns. Improvements were planned in order to obtain better answers in the presence of local discontinuities. Fusiello et al. [15] uses several local windows around the pixel. Devernay [5] uses a local window in form of parallelogram, and deforms it to obtain a minimum

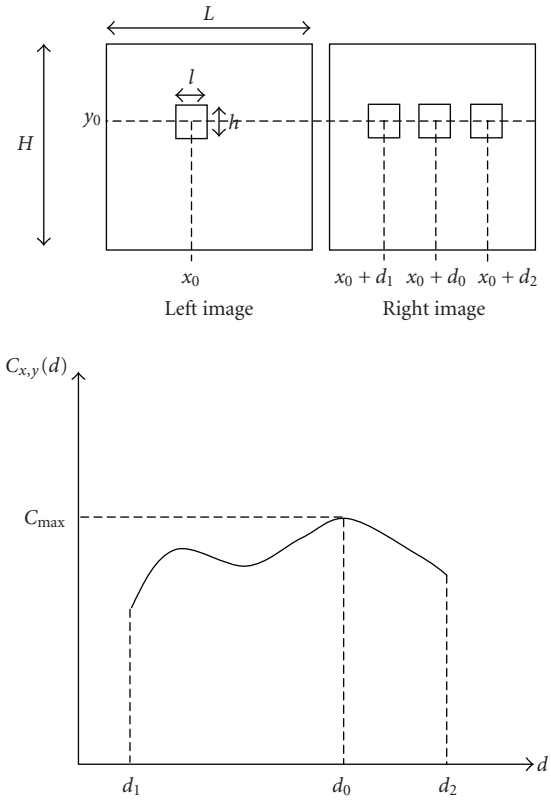


Figure 2: Crosscorrelation-based matching.

criterion. These two methods allow introducing local disparity gradients.

In our case, we improve the correlation results during a regularization step composed by a parabolic approximation of the correlation (allowing subpixel interpolation) and a morphological filtering which allows removing artifacts. The parabolic interpolation is given by

$$d(x,y) = d_0(x,y) + \frac{1}{2}\frac{C_{x,y}(d_0+1) - C_{x,y}(d_0-1)}{2C_{\max} - C_{x,y}(d_0+1) - C_{x,y}(d_0-1)}. \tag{3}$$

2.2.3. Local phase

The algorithm uses the image local phases estimates for the disparity determination [13]. Phase differences, phase derivative, and local frequencies are calculated by filtering the stereocouple with Gabor filters, as follows:

$$\begin{aligned} I_{1G}(x) &= I_1(x) * G(x,\sigma,\omega), \\ I_{2G}(x) &= I_2(x) * G(x,\sigma,\omega), \end{aligned} \tag{4}$$

with the Gabor kernel defined as

$$G(x,\sigma,\omega) = \frac{1}{\sqrt{2\pi\sigma}} e^{-x^2/(2\pi\alpha)^2}, \tag{5}$$

and the local phases are defined as

$$\begin{aligned} \Phi_1(x) &= \arctan\left(\frac{\mathrm{Im}\,(I_{1G}(x))}{\mathrm{Re}\,(I_{1G}(x))}\right), \\ \Phi_2(x) &= \arctan\left(\frac{\mathrm{Im}\,(I_{2G}(x))}{\mathrm{Re}\,(I_{2G}(x))}\right). \end{aligned} \tag{6}$$

The disparity d is calculated from estimates of local phases in images I_1 and I_2 using

$$d_\omega(x) = \frac{(\Phi_1(x) - \Phi_2(x))}{\overline{\omega}}, \tag{7}$$

where $\overline{\omega}$ is the average local spatial frequency.

The processing allows then to deduce local disparities [16]. The frequency scale limitations and the phase wrapping problem impose to limit the disparity. To obtain a higher range of disparity, it is necessary to resort to a coarse-fine strategy in which the results for each scale are extended and used on the following scale, thus making it possible to increase the limits of disparity variations [17]. A regularization step introduces a smoothing constraint for each scale by fitting the results to a spline surface. These methods are related to recent discoveries in physiology of three-dimensional perception [18, 19].

Another method based on local phase determination uses complex wavelets [20]. Through its robustness against lighting variation and additive noise, this method extends the properties of the Gabor wavelets to the differences in luminosity variation and to additive noise. But especially this operator provides shift invariance and a good directional selectivity. These conditions are essential to obtain disparity. The disparity computation is carried out by a difference between the detail coefficients of the left and right images. An adjustment by a least square method gives an optimal disparity, depending on the phase, and insensitive to intensity changes [21]. The epipolar constraint can be added effectively for a better determination of homologous points [22].

2.3. Methods comparison in the case of face acquisition

In order to choose a good compromise between performances and speed processing, we measured the quadratic error between a model of face and the stereo acquisition.

The error is defined as

$$Q = \frac{1}{LH}\sum_{y=1}^{H}\sum_{x=1}^{L} |\,O(x,y) - S(x,y)\,|^2, \tag{8}$$

where $O(x,y)$ represents the depth map obtained using our algorithms and $S(x,y)$ is the model depth map, obtained using a 3D laser-based scanner.

The face used for the comparison is depicted in Figure 3.

We studied the error depending on the focal length used during acquisition. We showed in [23] that the optimum choice for the stereo device depends on the focal length and that this optimum can be chosen around $f = 30$ mm for a standard CCD-based camera.

Rectified images of test face

Reference depth map

Figure 3: Reference face.

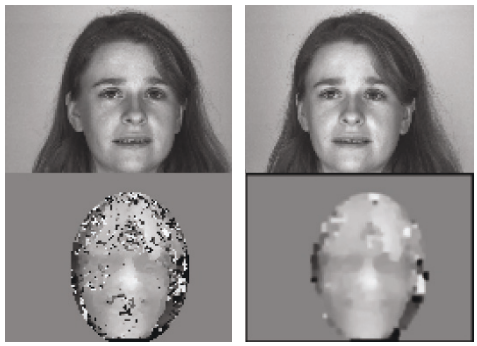
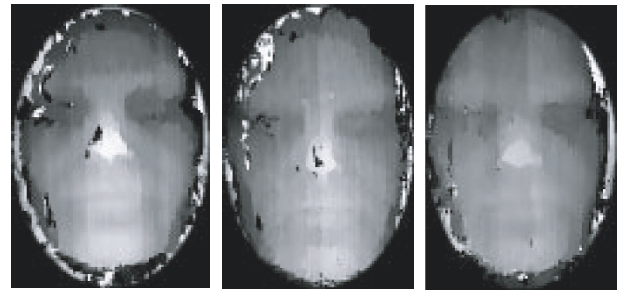


Figure 4: Left and right acquired images, depth maps without and with post processing.

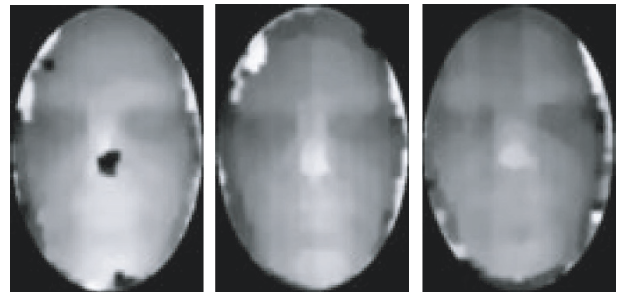
The maps obtained by crosscorrelation can be very correct, under certain conditions of illumination. For our part, we obtained good dense depth map by projecting a random texture on the face. Nevertheless, a post processing is required in order to effectively improve the existing discontinuities. This processing can be filled by a morphological opening and closing, followed by a Gaussian blur to smooth small discontinuities correctly. Figure 4 shows the results obtained with and without filtering.

The images obtained using the three compared methods are depicted on Figure 5, and the corresponding error is depicted on Figure 6. It is clear that, although the Gabor wavelets-based method seems to be the best choice, the performances are very close from each other when focal length is near $f = 30$ mm. This justifies our final choice of implementation, based on the crosscorrelation algorithm, for which the



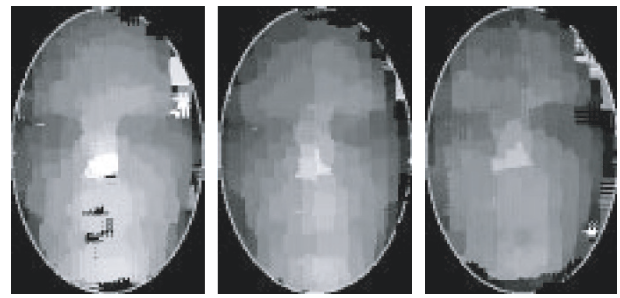
(I) $f = 28.5$ mm (II) $f = 32$ mm (III) $f = 35.3$ mm

(a) Results using crosscorrelation



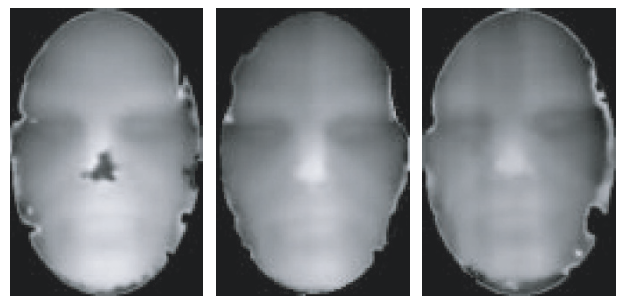
(I) $f = 28.5$ mm (II) $f = 32$ mm (III) $f = 35.3$ mm

(b) Results using filtered crosscorrelation



(I) $f = 28.5$ mm (II) $f = 32$ mm (III) $f = 35.3$ mm

(c) Correlation using multiple windows



(I) $f = 28.5$ mm (II) $f = 32$ mm (III) $f = 35.3$ mm

(d) Results using Gabor wavelets

Figure 5: Depth maps.

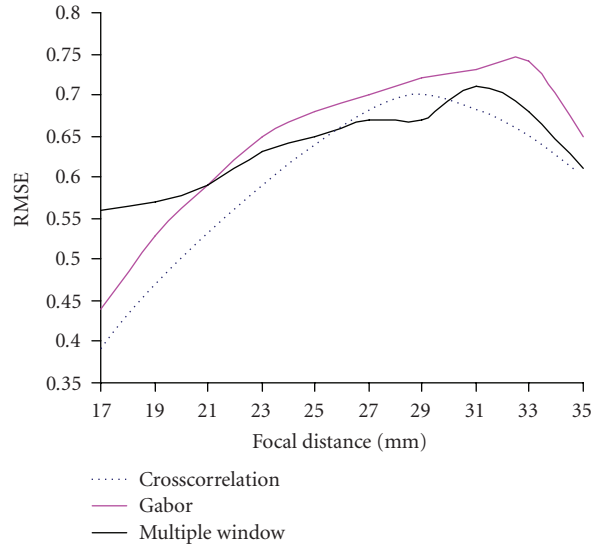


Figure 6: Error comparison.

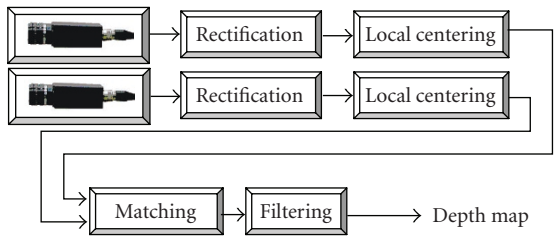


Figure 7: Computed chain implemented to obtain dense depth map.

hardware implement cost will be clearly lower than in the Gabor wavelets case.

3. PROCESSING, RESULTS, AND IMPLEMENTATION

Since we obtained using software simulation dense and precise depth maps, we implemented the crosscorrelation by shifting windows algorithm. The whole algorithm is distributed as shown in Algorithm 1, and is depicted in Figure 7.

In the first stage after image acquisition, we carry out an image rectification. This processing is computed through a weak calibration and the fundamental matrix determination [24]. The rectification matrices are obtained by an original computation, based on a projective method [25], by calculating the homography of four points of each image plan.

We carry out then a local centering of the images thus allowing reducing the problems involved in the average intensity differences between the two views. Data normalization does not produce more reliable results.

The following step is the two images matching, on a defined disparity range. This calculation is applied by a crosscorrelation by shifting windows. The used criterion is the difference absolute value sum (DAS). We sort then the values to seek their minimum.

(1) Acquisition of left and right images.
(2) Rectification of left and right images.
(3) Local centering of left and right images.
(4) Matching using crosscorrelation.
  (a) Crosscorrelation computation (2).
  (b) Disparity computation, using the search of maximum value of crosscorrelation and subpixel interpolation.
(5) Filtering of the depth map.

Algorithm 1

Table 1: Operation required.

	Number of operations
Rectification	$2 \times 4 \times L \times H$
Local centering	$2 \times 21 \times L \times H$
Matching-crosscorrelation	$21 \times L \times H \times D$
Matching-max determination	$(2 \times D_r + 3) \times L \times H$
Total	$(23 \times D_r + 53) \times L \times H$

We evaluated the number of operations to be performed in order to map the algorithm on an embedded architecture.

In order to realize a fast processing of the local centering and the crosscorrelation, we use an optimized computing algorithm described hereafter. Because of this algorithm, the number of operations we carry out is no more proportional to the crosscorrelation window size.

So, we have to compute the following values:

$$C_r(x) = C_r(x-1) - C(x-l) + C(x),$$
$$C_{rc}(x) = C_{rc}(x-1) + C_r(x) - C_r(x-hL), \quad (9)$$

where, C_r and C_{rc} are intermediate values, x represents the current computed value index and $x-1$ the previous index, h and l are the height and the width of the crosscorrelation window and L is the image width. The C_r and C_{rc} values are the results of a previous computing of the crosscorrelation. C_r is the value computed in an h pixels row wide, and C_{rc} is the value computed in an $h \times l$ pixels window. These values must be computed in real-time in order not to break the data flow.

The C_r and C_{rc} values are 16 bits coded and must be stored into arrays. The capacities of these needed arrays are for C_{rc}, the line width, and for C_r, the line width multiplied by the crosscorrelation window height. This processing is therefore a more important consumer of memory space than a crosscorrelation classical computing. Moreover, memories must be managed with a lot of consideration in order not to break the data flow of the whole processing.

We examine in Table 1 the number of operations we need to realize the different processing. The operations we use are elementary and include simple arithmetic operations

Table 2: Virtex devices.

Device	System gates	CLB array	Logic cells	Block RAM bits	Block RAM number
XCV300	322970	32×48	6912	65536	16
XCV800	888439	56×84	21168	114688	28

(addition or subtraction), incrementations of values for the loops and access memory operations.

In this table, H and L are the height and the width of the image and D_r the disparity range value. After some studies of this algorithm working on human faces [23], we determined the optimal values for the crosscorrelation window size and the disparity range. We use a 256×256 pixels image size, a 7×6 pixels crosscorrelation window size and 20 for the disparity range. The number of operations we have to compute is then equal to 33, 62 Mops per trame, or 840 Mops per second for a 25-image-per-second video standard.

In order to optimize the implementation of the steps 1, 2, 3, and 4a in Algorithm 1 using parallel computing, we choose to use a reconfigurable logical device.

These processings are carried out effectively on the XILINX FPGA Virtex. Virtex devices provide better performance than previous generations of FPGA. Designs can achieve synchronous system clock rates up to 200 MHz including for Inputs-Outputs. Virtex devices feature a flexible regular architecture that comprises an array of configurable logic blocks (CLB) surrounded by programmable input/output blocks (IOBs), all interconnected by a hierarchy of fast and versatile routing resources. They incorporate also several large blocks RAM memories. Each block RAM is a fully synchronous dual-ported 4096-bit with independent control signal for each port. The data widths of the two ports can be configured independently. Thus, each block has 256 datas of 16- bit capacity. Each memory blocks are organized in columns. All Virtex devices contain two such columns, one along each vertical edge. The Virtex XCV300 and XCV800 capacities are grouped together in Table 2.

An original parallel implementation, described in the next paragraph, allows a very fast calculation of the criteria on all the disparity range.

These results are then given to a DSP which carries out successively the following processing: a parabolic interpolation to obtain wider disparity values; morphological filtering made up of an opening then a closing to eliminate wrong disparities while keeping depth map precision; a Gaussian blurring filter finally to smooth the obtained results. These processings are optimized on a C6x Texas Instrument DSP which allows a fast data processing.

3.1. Description of the chosen architecture

The constraints imposed by the algorithmic sequence real-time computing and the needed compactness to obtain an embedded architecture lead us to choose a reconfigurable and multiprocessor FPGA-DSP set: the Mirotech Arix Board. This board is designed in several independent computing parts, with configurable links. External links allow us to interface the board with a real-time Frame Grabber (FG) and with a PC (through the PCI bus). The computing parts are as follows (Figure 8): one virtual processing element (VPE) consisting of a Xilinx Virtex FPGA (XCV300 or XCV800) with four 512ko SBSRAM memory blocks; the second is composed of one Texas Instrument TMS320C44 DSP with two 1Mo SRAM memory blocks. This DSP interfaces two TIM sites on which we can connect the third computing element. For this part, we choose one Texas Instrument TMS320C67 DSP with an 8Mo SDRAM memory block. These three parts are connected by configurable links that allow direct memory access (DMA). Thus whole processing can be done in pipeline, cascaded in several parts as FG⇒VPE⇒DSPC44⇒DSPC67⇒DSPC44⇒PCI.

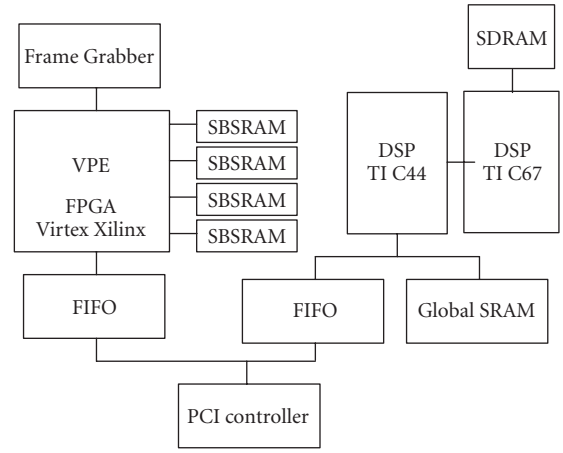


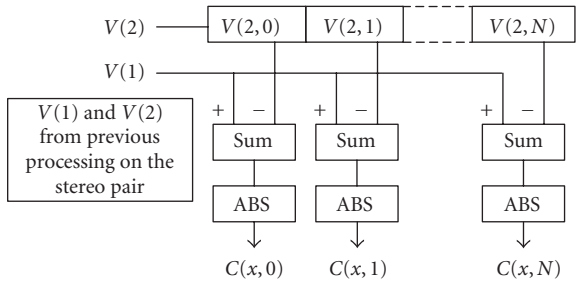
Figure 8: Parts of the board architecture.

This reconfigurable architecture allows us to quickly realize and validate our algorithm-architecture suitability.

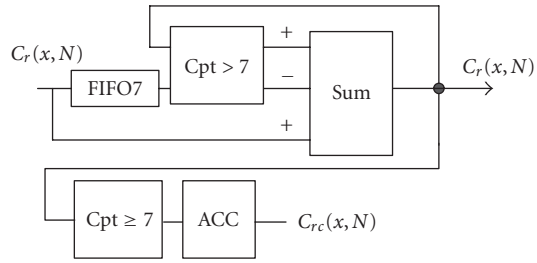
3.2. Matching implementation

The most important computation time is required by the matching processing; so we made a particular effort to implement this part. To obtain real-time results, we use the optimised crosscorrelation technique implemented using the intrinsic parallelism of FPGA.

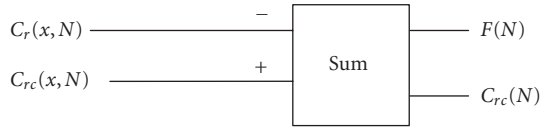
This method, described in a previous section, allows an important time gain by reusing intermediary computed results. Although a C language implementation of this algorithm is relatively simple, its FPGA implementation presents more problems. The main is memory management. Indeed, this processing needs a lot of intermediary values, easily allocated in C on a PC. Unfortunately, in order to respect the real-time constraint, we have to reduce the memory access and manage the best possible intermediary values and the data flow. Three processing parts are implemented: the first (Figure 9(a)) for the DAS parallel processing on the disparity



(a) DAS computing



(b) Intermediate values computing



(c) Final values computing

Figure 9: Matching implementation.

range; the second (Figure 9(b)), for the intermediary values parallel computing, and the third (Figure 9(c)) for the final computing. These two first parts hold, respectively, 9% and 6% slices of a Virtex300 FPGA.

For the parallel processing, we connect N times (N is between 0 and 19) the second part to all the outputs of the first part. We obtain thus in parallel the whole criteria needed to compute the disparity for one pixel. The C_{rc} criteria are stored in the Virtex memory blocks at the rate of one memory block per disparity. The C_r criteria are alternately stored into two SBSRAM blocks of the Arix board. For each even line, the writing is carried out into the first block and, for the odd lines, into the second block. The two memory blocks can then be used in parallel. This allows processing the third part, in which a reading of the C_{rc} and C_r criteria is carried out, without any influence onto the two other parts.

The whole final criteria, named $F(N)$, are then used for the determination of the maximum disparity onto the disparity range. The maximum disparity is determined, and we keep, with this value, the previous and the following disparity values. These three values are then sent to the DSP (which is well adapted to floating point processing) for a subpixel determination of the disparity (a parabolic interpolation, according (3)).

4. CONCLUSIONS AND PERSPECTIVES

We compared in the present paper various stereo matching methods in order to study real-time 3D face acquisition. We have shown that it is possible to implement a simple crosscorrelation-based algorithm with good performances, using post processing. A various architecture study led us to a judicious choice allowing obtaining the desired result. The real-time data processing is implemented on an embedded architecture. We obtain a dense face disparity map, precise enough for considered applications (multimedia, virtual worlds, biometrics) and using a reliable method. In particular, we plane to use the results as features for a face recognition software described in a previous article [26].

REFERENCES

[1] C. Beumier and M. Acheroy, "Automatic face verification from 3D and grey level clues," in *Proceedings of the 11th Portuguese Conference on Pattern Recognition (RECPAD '00)*, pp. 95–101, Porto, Portugal, May 2000.

[2] T. S. Jebara and A. Pentland, "Parametrized structure from motion for 3D adaptive feedback tracking of faces," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pp. 144–150, San Juan, Puerto Rico, USA, June 1997.

[3] J. Y. Cartoux, *Formes dans les images de profondeur. Application à la reconnaissance et à l'authentification de visages*, Ph.D. thesis, Université Blaise Pascal, Clermont-Ferrand Cedex, France, 1989.

[4] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[5] F. Devernay, *Vision stéréoscopique et propriétés différentielles des surfaces*, Ph.D. thesis, Ecole Polytechnique, l'Institut National de Recherche en Informatique et en Automatique, Chesnay Cedex, France, 1997.

[6] O. Faugeras, B. Hotz, H. Mathieu, et al., "Real time correlation based stereo: algorithm implementations and applications," Tech. Rep. RR-2013, l'Institut National de Recherche en Informatique et en Automatique, Chesnay Cedex, France, 1993.

[7] J.-R. Ohm and E. M. Izquierdo, "An object-based system for stereoscopic viewpoint synthesis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 5, pp. 801–811, 1997.

[8] B. Porr, A. Cozzi, and F. Wörgötter, "How to "hear" visual disparities: real-time stereoscopic spatial depth analysis using temporal resonance," *Biological Cybernetics*, vol. 78, no. 5, pp. 329–336, 1998.

[9] A. Koschan, "What is new in computational stereo since 1989: a survey of current stereo papers," Technischer Bericht 93-22, Technische Universiteät Berlin, Berlin, Germany, 1993.

[10] U. R. Dhond and J. K. Aggarwal, "Structure from stereo—a review," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no. 6, pp. 1489–1510, 1989.

[11] L. Alvarez, R. Deriche, J. Sanchez, and J. Weickert, "Dense disparity map estimation respecting image discontinuities," Tech. Rep. 3874, l'Institut National de Recherche en Informatique et en Automatique, Chesnay Cedex, France, 2000.

[12] M. R. M. Jenkin and A. D. Jepson, "Recovering local surface structure through local phase difference measurements," *CVGIP: Image Understanding*, vol. 59, no. 1, pp. 72–93, 1994.

[13] P. Fua, "A parallel stereo algorithm that produces dense depth maps and preserves image features," *Machine Vision and Applications*, vol. 6, no. 1, pp. 35–49, 1993.

[14] R. Deriche and O. Faugeras, "Les EDP en traitement des images et vision par ordinateur," *Traitement du Signal*, vol. 13, no. 6, 1996.

[15] A. Fusiello, V. Roberto, and E. Trucco, "Efficient stereo with multiple windowing," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pp. 858–863, San Juan, Puerto Rico, USA, June 1997.

[16] M. W. Maimone and S. A. Shafer, "Modeling foreshortening in stereo vision using local spatial frequency," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '95)*, vol. 1, pp. 519–524, Pittsburgh, Pa, USA, August 1995.

[17] J. Hoey, "Stereo disparity from local image phase," Tech. Rep., University of British Columbia, Vancouver, British Columbia, Canada, June 1999.

[18] I. Ohzawa, G. C. DeAngelis, and R. D. Freeman, "The neural coding of stereoscopic depth," *NeuroReport*, vol. 8, no. 3, pp. 3–12, 1997.

[19] P. Churchland and T. Sejnowski, *The Computational Brain*, MIT Press, Cambridge, Mass, USA, 1992.

[20] N. Kingsbury, "Image processing with complex wavelets," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 357, no. 1760, pp. 2543–2560, 1999, on a discussion meeting on "wavelets: the key to intermittent information?", London, UK, February 1999.

[21] H. Pan and J. Magarey, "Phase-based bidirectional stereo in coping with discontinuity and occlusion," in *Proceedings of International Workshop on Image Analysis and Information Fusion*, pp. 239–250, Adelaide, South Australia, November 1997.

[22] J. Magarey, A. Dick, P. Brooks, G. N. Newsam, and A. van den Hengel, "Incorporating the epipolar constraint into a multiresolution algorithm for stereo image matching," in *Proceedings of the 17th IASTED International Conference on Applied Informatics*, pp. 600–603, Innsbruck, Austria, February 1999.

[23] J.-P. Zimmer, "Modélisation de visage en temps réel par stéréovision," Thesis, University of Burgundy, Dijon, France, 2000.

[24] Z. Zhang, "Determining the epipolar geometry and its uncertainty: a review," *International Journal of Computer Vision*, vol. 27, no. 2, pp. 161–195, 1998.

[25] R. I. Hartley, "Theory and practice of projective rectification," *International Journal of Computer Vision*, vol. 35, no. 2, pp. 115–127, 1999.

[26] J.-P. Zimmer, J. Mitéran, F. Yang, and M. Paindavoine, "Security software using neural networks," in *Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (IECON '98)*, vol. 1, pp. 72–74, Aachen, Germany, August-September 1998.