https://doi.org/10.2352/ISSN.2470-1173.2018.11.IMSE-447
© 2018, Society for Imaging Science and Technology

# A near pixel Depth from Focus architecture for video rate depth estimation

***Simon Emberger, Laurent Alacoque, Antoine Dupret, Gilles Sicard ; Univ. Grenoble Alpes, CEA, LETI, Grenoble, France***
***Jean Louis de Bougrenet de la Tocnaye ; Optics department ; IMT-Atlantique; Brest ; France***

## Abstract

*Among the various techniques that allow the acquisition of the depth of the scene, Depth from Focus (DfF) technique is a good candidate for low-resources real-time embedded systems. Indeed it relies on low complexity processing and requires one single camera. On the other hand, the large data dependency imposed by the size of a focus-cube must be tackled in order to ensure the embeddability of the algorithm. This paper presents algorithm improvement and an architecture optimized for both processing complexity and memory footprint. For full-HD images, this architecture can produce depth and confidence maps in real time using roughly 1.4p arithmetic operations per pixel, where p is the number of depth planes, without the need of a multiplier, while the needed memory footprint is equivalent to 6% of one frame. All in focus images can also be processed on-the-fly to the price of an additional 2 frames memory buffer.*

## Introduction

With the development of new autonomous systems, smart image sensors are being designed, and algorithms simplified, in order to provide more relevant information about the near environment. The depth of the scene is a crucial information for environment sensing, and many works provide solutions to capture it. Indeed, for an autonomous vehicle, knowing the distance to surrounding objects is of primary importance to preserve its safety and its integrity.

Most techniques that have been considered to measure the depth of the scene, require extra hardware e.g. light sources in active Time of Flight imaging [1] or more than a single camera for stereovision [2-5, 12]. Democratization of autonomous systems leads scientists to find more compact and affordable ways to get object distances from images. Retrieving the depth of the scene with a low cost system requires the combination of low cost optics and a compact image sensor. When cutting down the costs of a system is the main stake, embedding depth extraction processing as close to the pixel array within a single chip is the only viable alternative. Although at the top of the integration and power, phase-detection pixels [6] and Depth from Defocus [7] techniques offer new hardware and software solutions for depth evaluation. Yet such techniques require considerable post processing involving a lot of computational cost. Other techniques deduce depth maps from the sharpness variations of images captured during a focus sweep (depth from focus: DfF) [8-11]. These techniques have the advantages of requiring limited computational power and a single standard camera with a variable focus lens. However it often needs more memory than stereoscopic solution. Also the output depth map is most of the time filtered with a threshold and only a sparse depth map of the scene is available [11]. In fact, with the emerging data fusion systems, it becomes more interesting to provide a full depth map and associated relevance coefficients [12]. Meaningful relevance coefficients help to reduce false alarm rates and thus increase the safety of the system. Our previous work presented in [8] aimed to provide a full depth map along with relevance coefficients, referred to in this paper as confidence map. This confidence map results from the self-evaluation of the depth measurement quality for each depth pixel. A high confidence index indicates a more trustable depth estimation. This work presents an implementation architecture of the DfF method proposed in [8] in order to provide RGB-D-C pixels where RGB is an all in focus colored pixel, D its depth and C the confidence value used to weight the depth value.

In the following, the next section presents a brief description of the algorithm presented in [8] with some improvements designed to limit the memory footprint. Then a hardware architecture for low-complexity and low memory footprint implementation of this depth estimation algorithm is proposed. Finally some results of simulation are shown.

## Depth extraction algorithm and all in focus imaging

As previously stated, in near-pixel embedded algorithms, the processing must be kept compact in order to limit silicon area for the image sensor, hence keeping its cost low. In this paper, we briefly review the algorithm presented in [8], and propose improvements and a hardware architecture for low-memory-footprint and low-complexity implementation.

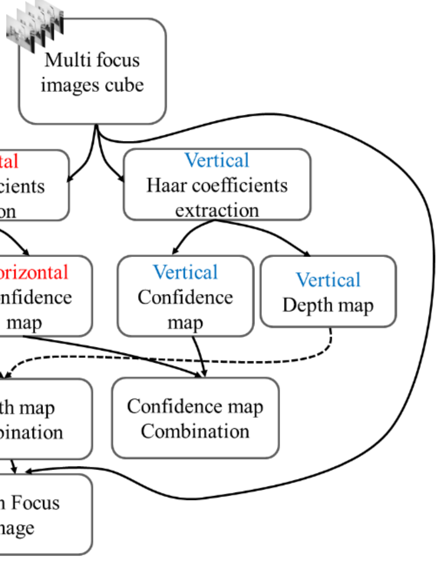
Multi focus images cube
Horizontal Haar coefficients extraction
Vertical Haar coefficients extraction
Horizontal Depth map
Horizontal Confidence map
Vertical Confidence map
Vertical Depth map
Depth map Combination
Confidence map Combination
All in Focus Image

*Figure 1. Schematic representation of the algorithm principle*

Figure 1 presents the algorithm proposed in [8]. It is based on a bidirectional contrast analysis. Two depth and two confidence maps are extracted and then combined in order to provide two unified depth and confidence maps. As stated before, the purpose of the confidence map is to weight the depth map with confidence values in order to predict for every depth pixel how good its evaluation is.

DfF relies on images with shallow depth of field to improve the depth extraction sensibility. This in turn strengthens the blur of out of focus zones which reduces their usability for image analyses. So an additional all in focus (AiF) image of the scene is computed.

The aforementioned reasons of silicon cost impose the limiting of the memory footprint to its lowest end. DfF requires a cube of images taken at several foci. The greater the depth resolution, the larger the focus cube and its impact on the memory budget. The architecture presented in the next section was designed to keep the memory footprint as low as possible while maintaining sufficient quality. For this, the algorithm proposed in [8] is adapted to use adjacent nonoverlapping windows processing instead of overlapping (sliding) window. It reduces the quantity of data which must be buffered during the process. RAW images with varying focus are captured to form a cube of images. Since monochrome images are more suitable for contrast analyses, the Bayer pattern is first removed using 2×2 binning. The local sharpness within each resulting image is then evaluated by computing the sharpness criterion given in (1).

$$S_{h/v}(x,y,z) = \max_{X,Y}(H_{h/v}(X,Y,z)) - \min_{X,Y}(H_{h/v}(X,Y,z)) \quad (1)$$

where $H_{h/v}$ represents the horizontal and vertical 2D Haar wavelet coefficients of the image with focus index $z$ in the multi-focus cube. $(X, Y)$ are the coordinates of coefficients included in the window of size $w_w \cdot w_h$ centered on $(x,y)$. In the following, depth map refers to the matrix of focus indexes. Actual distance can easily be computed from the focus index using the classic geometrical optic equations as described in [11].

From the two sharpness criterion arrays $S_{h/v}$, two depth and two confidence maps, respectively $DM_h$, $DM_v$, $C_h$ and $C_v$ are extracted using (2) and (3) respectively.

$$DM_{h/v}(x,y) = \arg\max_{z}\left(S_{h,v}(x,y,z)\right) \quad (2)$$

$$C_{h/v}(x,y) = \max_{z}(S_{h/v}(x,y,z)) - \min_{z}(S_{h/v}(x,y,z)) \quad (3)$$

Both the depth and the confidence maps are then combined to form a unified depth map $DM$ and its associated confidence map $C$ using respectively (4) and (5).

$$DM(x,y) = \begin{cases} DM_h(x,y) & if \quad C_v(x,y) \le C_h(x,y) \\ DM_v(x,y) & if \quad C_v(x,y) > C_h(x,y) \end{cases} \quad (4)$$

$$C(x,y) = \begin{cases} C_h(x,y) & if \quad C_v(x,y) \le C_h(x,y) \\ C_v(x,y) & if \quad C_v(x,y) > C_h(x,y) \end{cases} \quad (5)$$

Finally, an $AiF$ image is computed according to (6).

$$AiF(x,y) = I(x,y,uDM(x,y)) \quad (6)$$

where $I$ represents the multi-focus cube of source images and $uDM$ the up-sampled depth map with the same resolution as the source images. Each $AiF$ pixel value corresponds to the pixel value of the image in the focus cube indexed by the depth map.

As it can be seen from the equations (1-6), the algorithm proposed in [8] relies on simple arithmetic operators such as adders and comparators. The nonoverlapping windows implementation proposed in this paper further reduces the computation power needed to implement it at the expense of an acceptable loss of quality as it will be shown later in this paper. To guarantee the success of depth estimation, the scene must not change during the acquisition of the focus cube. Hence a fast acquisition is required. According to human cinematic experiments, a minimum depth frame rate of 16fps is required to perceive continuous motion. With this frame rate, an application with 10 depth levels needs to acquire RAW images as fast as 160fps. For image processing faster frame rate might be useful and the pipelined architecture proposed here gracefully handles scalability thanks to its low processing requirements. Additionally, as mentioned above, the required memory must be drastically kept low in order to maintain an acceptable cost. The architecture presented below was designed with both constraints in mind.

In the following, the width and the height of a RAW image are referred to as $I_w$ and $I_h$. $w_w$ and $w_h$ correspond to the width and the height of the nonoverlapping windows, $nb_x$ and $nb_y$ are the number of windows along the two directions as defined by (7).

$$nb_x = \frac{I_w}{2 \cdot 2 \cdot w_w} \qquad nb_y = \frac{I_h}{2 \cdot 2 \cdot w_h} \quad (7)$$

The numbers 2 in the denominator denote the scale reduction due to the binning and 2D Haar coefficient computing.

## Hardware Architecture

The proposed architecture overview is shown on Figure 2.

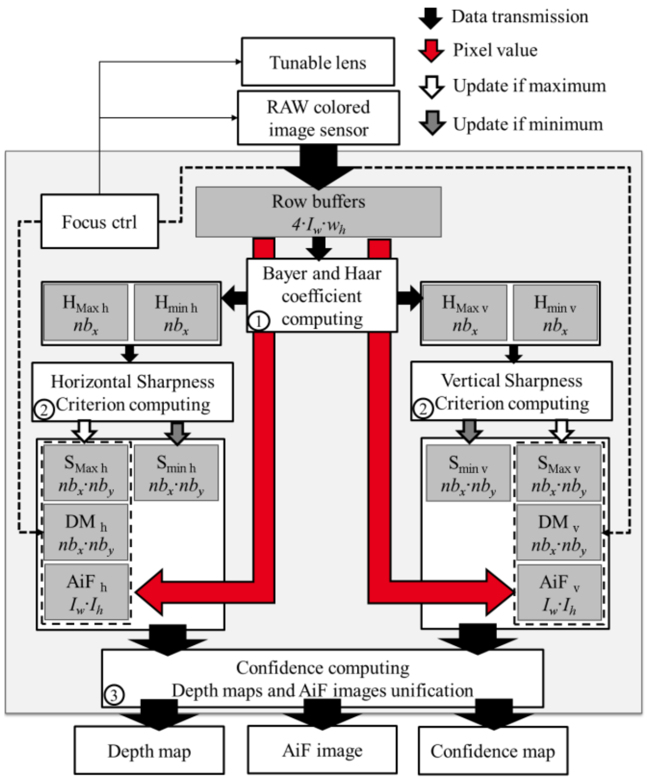
Data transmission
Pixel value
Update if maximum
Update if minimum
Tunable lens
RAW colored image sensor
Focus ctrl
Row buffers $4 \cdot I_w \cdot w_h$
Bayer and Haar coefficient computing
1
$H_{Max\,h}$ $nb_x$
$H_{min\,h}$ $nb_x$
$H_{Max\,v}$ $nb_x$
$H_{min\,v}$ $nb_x$
2 Horizontal Sharpness Criterion computing
2 Vertical Sharpness Criterion computing
$S_{Max\,h}$ $nb_x \cdot nb_y$
$S_{min\,h}$ $nb_x \cdot nb_y$
$DM_h$ $nb_x \cdot nb_y$
$AiF_h$ $I_w \cdot I_h$
$S_{min\,v}$ $nb_x \cdot nb_y$
$S_{Max\,v}$ $nb_x \cdot nb_y$
$DM_v$ $nb_x \cdot nb_y$
$AiF_v$ $I_w \cdot I_h$
3 Confidence computing Depth maps and AiF images unification
Depth map
AiF image
Confidence map

*Figure 2. Architecture for depth map, confidence map and all in focus image generation. Gray rectangles represent required memory blocks whose sizes are indicated with italic style.*

It is composed of three main processing stages marked with circled numbers on Figure 2. These processing stages are data dependent but run in parallel. The first stage prepares the sharpness criterion extraction as described by (1). The second corresponds to the computing of the criterion along the two main directions. It also extracts the depth evaluations and updates the $AiF$ images as described by (2) and (6). The final outputs, described by (4) and (5), are computed in the third and last processing stage. Required memory is represented with gray blocks along with its required size in words.

A timing representation of the processing flow is proposed Figure 3.

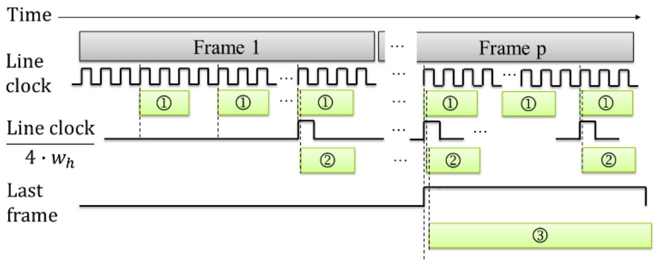

Figure 3. Functional processing timing diagram.

## Pre-processing for the sharpness criterion

In a first stage ① on Figure 2, Bayer pixels are converted into a lower resolution monochrome image. For this, 2×2 binning is performed on Bayer pixels, thus subsampling the image with a 4 times factor and saving memory. The monochrome conversion occurs for each four rows acquisition of the source RAW image. It begins while the fourth line starts to be saved in the row buffer.

Then the horizontal and vertical Haar wavelet coefficients are computed. For each 2×2 monochrome pixel block, it requires 3 adders and one 2 bits right shifts to compute a Haar coefficient. The process therefore requires 2 rows of monochrome image, hence 4 RAW rows of an original image. For these two operations, the minimum size of the input line buffer must hence be of $4 \cdot I_w$ words. However, because of the *AiF* image generation, the input line buffer must have a size of at least $4 \cdot I_w \cdot w_h$ words. Indeed, the *AiF* image cannot be updated until a full $w_w \cdot w_h$ window is entirely processed. Corresponding pixels must hence be kept in memory until that time. For each window, the two extrema of the Haar coefficients along the horizontal and vertical directions are recorded in order to be used for the sharpness criterion computation later in the process. This requires four additional buffers of $nb_x$ words each. Each new $H_x$, $H_y$ coefficient is compared to the previously stored extrema of the same window and saved if it is respectively bigger or smaller than the previous recorded maximum or minimum as shown on Figure 4.

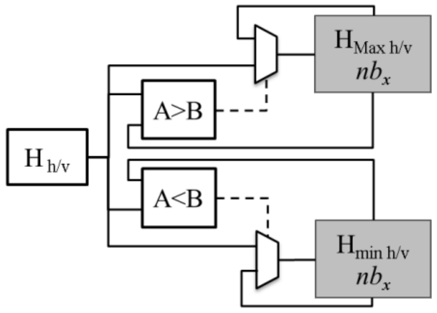

Figure 4. Architecture for the pre-processing of the sharpness criterion: memorization of Haar coefficient extremums. Gray rectangles represent required memory blocks whose sizes are indicated with italic style.

Ones can notice that if colored images are not needed, a monochrome sensor might be used. It would reduce the processing cost by avoiding the binning operation. Also the input buffer might have a size of $2 \cdot I_w \cdot w_h$ words.

## Sharpness criterion computing and temporary all in focus images and depth maps generation

The computing of the criterion described by (1) occurs each time a window has been processed (② on Figure 2). Since the confidence map is calculated as the dynamic range of the criterion, the criterion extrema must be saved for each window, in the two directions in four memory blocks of $nb_x \cdot nb_y$ words each. They will be used to generate the confidence map (3) during the final stage.

Every time the criterion maxima matrices are updated, the corresponding temporary depth map is updated with the focus index of the currently processed image. To save these depth maps, two additional memory blocks of $nb_x \cdot nb_y$ words are used.
To limit the memory footprint, the focus cube is never entirely recorded as it was the case in [8]. However, since the *AiF* image can be a combination of any image of the focus cube, some part of it are saved during the process into two *AiF* image buffers, one for each direction. Each is updated when the maximum of the criterion along the corresponding direction increases. Two full frame memories are required to save these two *AiF* images. The whole processing is shown Figure 5.

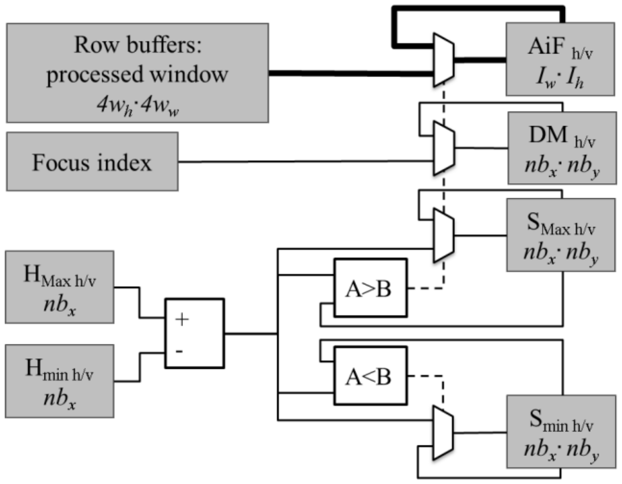
Row buffers: processed window $4w_h \cdot 4w_w$
Focus index
$H_{Max\ h/v}$ $nb_x$
$H_{min\ h/v}$ $nb_x$
+
-
A>B
A<B
$AiF_{h/v}$ $I_w \cdot I_h$
$DM_{h/v}$ $nb_x \cdot nb_y$
$S_{Max\ h/v}$ $nb_x \cdot nb_y$
$S_{min\ h/v}$ $nb_x \cdot nb_y$

Figure 5. Architecture for the sharpness criterion computing and the AiF image and depth map memory updates. Gray rectangles represent required memory blocks whose sizes are indicated with italic style.

## Unified depth map, all in focus image and confidence map extraction

At every end of a focus sweep cycle, the confidence values of the two directions are computed in parallel as the dynamic range of the sharpness criterion and then compared (③ on Figure 2). The unified confidence map is composed of the highest confidence indexes in each directions. *AiF* blocks and depth estimations that correspond to the direction with the highest confidence are selected at the same time to form the unified depth map and the *AiF* image. For each pixel, the *AiF* RAW data, the depth estimation and the confidence index are then transmitted on the fly to the host system on 3 distinct output channels. The whole process is represented on Figure 6.

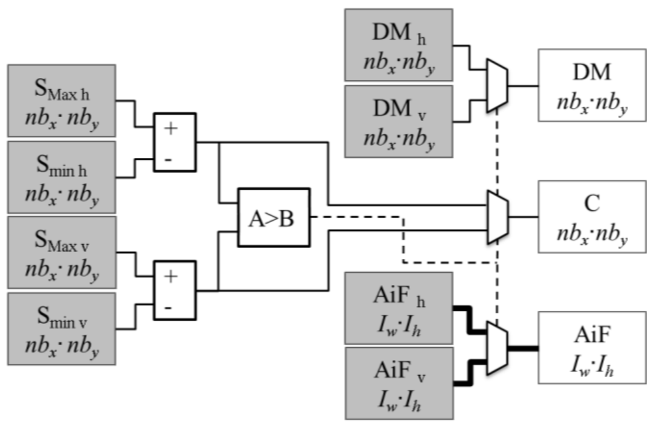

*Figure 6. Architecture for unified depth map, AiF image and unified confidence map generation. Gray rectangles represent the memory blocks. Gray rectangles represent required memory blocks whose sizes are indicated with italic style*

## Results

To compare the results of the sliding method and the nonoverlapping one, the experimental dataset of [8] acquired with a Nikon D300S with a Nikkor AF-S DX 16-85 mm f/3.5-5.6G ED VR camera lens. It is composed of three scenes forming three multi-focus cubes of 14 images. The Figure 7 shows these scenes and their corresponding manually designed ground-truth depth map. In the following, the first scene is referred to as 'sun', the second as 'panda' and the third as 'chair'.

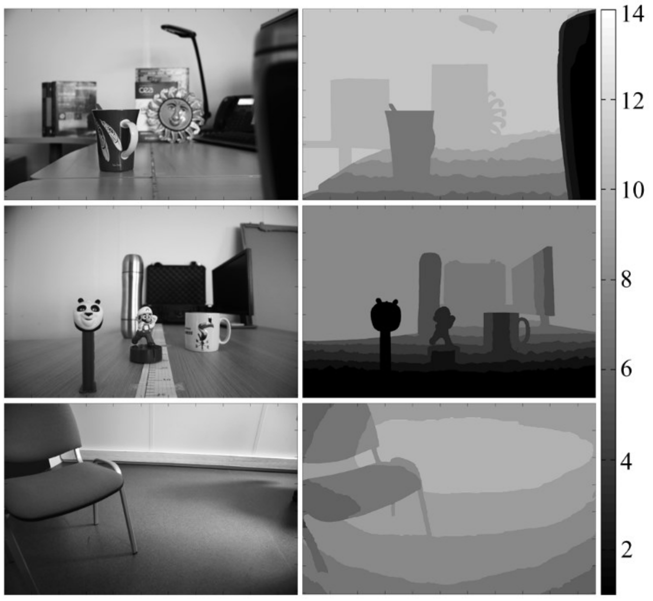

*Figure 7. Experimental scenes (left) and the corresponding ground-truth depth maps with their depth index scale (right). The darker, the nearer.*

Some results from this algorithm are compared on Figure 8, Figure 9 and Figure 10 using both sliding window [8] (left) and nonoverlapping windows (this work, right).

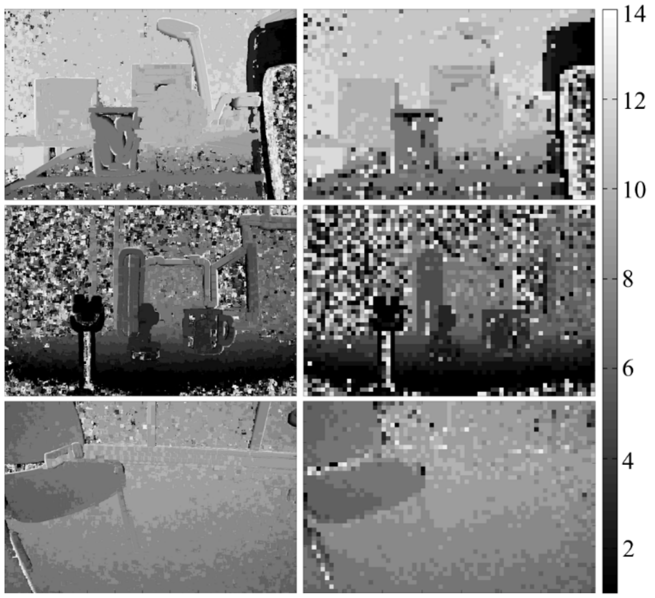

*Figure 8. Comparison of depth maps generated using the sliding window version [8] (left) and nonoverlapping windows version (this work, right) for 15 depth levels. The darker, the nearer. $w_x$=$w_y$ =16.*

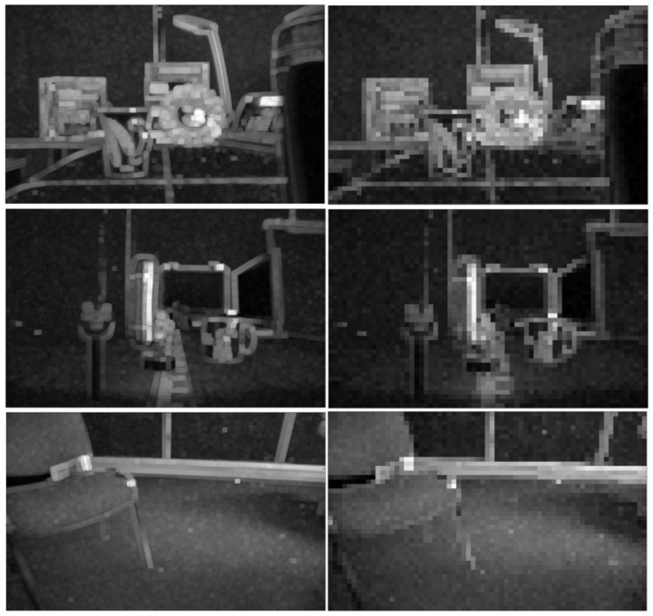

*Figure 9. Comparison of confidence maps generated using the sliding window version [8] (left) and nonoverlapping windows version (this work, right) for 15 depth levels. The lighter the better the estimation. $w_x$=$w_y$ =16.*

The depth maps on Figure 8 and confidence maps on Figure 9 show that besides the loss of resolution, most of the depth information remains with the distinct windows approach. Table 1 shows the correct-estimation-count ratio introduced in [8]. It represents the ratio of correct depth estimations with an index error tolerance of 1 depth plane in comparison with the ground truth over the pixel total count.

**Table 1. Comparison of the two depth extraction methods with the ground truth using correct-estimation-count ratio given in % (the higher, the better).**

| Image | Sun | Panda | Chair |
|---|---|---|---|
| Sliding [8] | 80.56 | 61.93 | 93.30 |
| Nonoverlapping | 79.60 | 60.38 | 92.77 |

As it can be seen, the correct estimated ratios of the proposed methods are really close to the ones evaluated in the case of the more computationally-intensive sliding window approach [8].

Qualitative results are presented on Figure 10, we notice that the *AiF* images from the two methods provide quite similar results even if those generated with the nonoverlapping windows processing exhibit a few more artifacts (see, e.g., the red rectangle on Figure 10). Yet, some elements in these images look sharper with the non-overlapping processing (see, e.g., the orange rectangle on Figure 10).

We find that this method is of sufficient quality for fast, computing-efficient depth estimation and all-in-focus image rendering for a great number of applications.

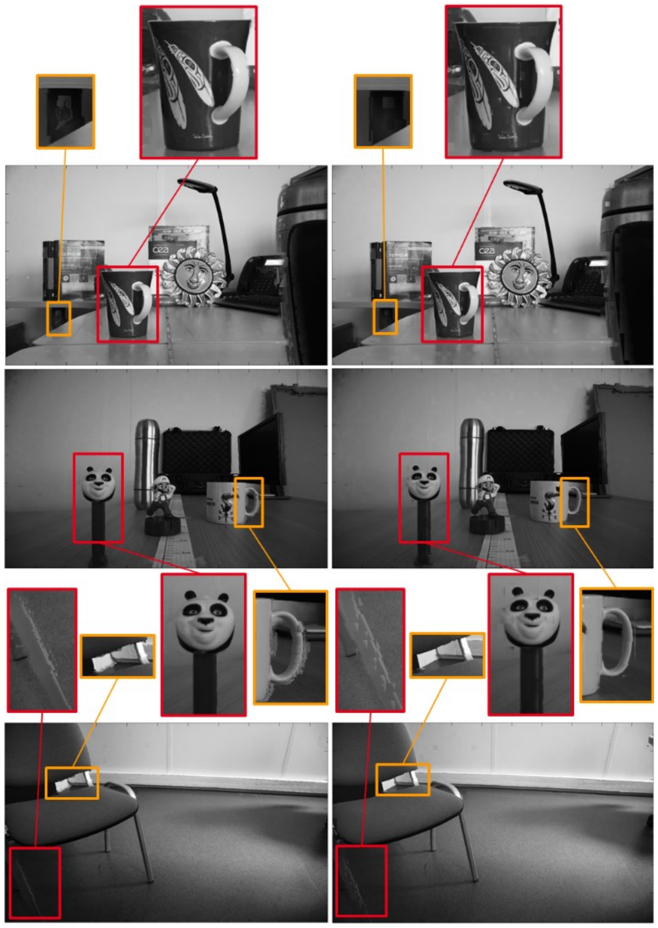
*Figure 10. Comparison of AiF images generated using the sliding windows version [8] (left) and nonoverlapping windows version (this work, right) for 15 depth levels. $w_x$=$w_y$ =16.*

## Memory footprint

The total needed memory $M$ for the process with the nonoverlapping windows can be estimated by equation (8). It should be noted that the memory size does not depend on the depth resolution. This allows this architecture to dynamically adjust the depth resolution at run time to adapt to the requirements of various applications.

$$M = 4 I_w w_h + \frac{I_w}{4 w_w}\left(4 + 6 \frac{I_h}{4 w_h}\right) + 2 I_w I_h \qquad (8)$$

Also, if the *AiF* image is not necessary, the last term *$2I_wI_h$* of the equation is no more useful. It reduces drastically the memory footprint.

As example, we considered a cube of 14 images with a resolution of 1080×1920 and 16 by 16 pixels windows. Comparisons of memory footprints between four architectures are presented Table 2. The two window processing methods (sliding and nonoverlapping) were compared, with and without the *AiF* image generation.

As it can be seen, the nonoverlapping architecture allows to save up to 96% of the memory compared to the sliding window architecture and up to 98.5% of the memory with the solution proposed in [11] when *AiF* images are not computed.

**Table 2. Comparison of four architectures (*$w_h$*=*$w_w$*=16)**

| | Memory (frames) |
|---|---|
| Sliding window with AiF generation [8] | 3.45 |
| Nonoverlapping with AiF generation (this work) | 2.06 |
| Method from [11] with AiF generation | 5.00 |
| Nonoverlapping without AiF generation (this work) | 0.06 |
| Method from [11] without AiF generation | 4.00 |
| Sliding window without AiF generation [8] | 1.45 |

## Processing complexity

The number of 2 input comparators and adders needed to compute a full resolution RGB-D-C pixel was evaluated. It is shown in Table 3 where $p$ represents the number of wanted depth planes.

**Table 3. Number of operations per pixel needed for each output pixel-depth-confidence**

| Architecture | Adders | Comparators |
|---|---|---|
| Nonoverlapping | $\frac{9}{8} \cdot p + \frac{p+1}{8 \cdot w_h \cdot w_w}$ | $\frac{p}{4} + \frac{4p+1}{16 \cdot w_h \cdot w_w}$ |
| Sliding [8] | $\frac{5}{4} \cdot p + \frac{1}{8}$ | $\frac{p}{4} + \frac{p \cdot w_h \cdot w_w + 1}{4}$ |

Since comparisons are generally designed using the sign bit of a substractor, the total number of operation can be summed up as a number of additions per pixel. This is summarized in Table 4 when $w_h=w_w=w$.

**Table 4. Total number of additions per pixel needed for each output pixel-depth-confidence**

| Architecture | Number of operations ($w_h=w_w=w$) |
|---|---|
| Sliding | $\frac{p}{4}(w^2+6)+\frac{3}{16} \underset{w^2>>p}{\approx} \frac{p}{4}(w^2+6)$ |
| Nonoverlapping | $\frac{11p}{8}+\frac{6p+3}{16\cdot w^2} \underset{w^2>>p}{\approx} p\cdot\frac{11}{8}$ |

As it can be seen, roughly *1.4p* arithmetic operations per pixel are needed with the proposed method whatever the image resolution.

In the same conditions as in the example of the previous section ($w=16$ and $p = 14$), the cumulated number of additions per pixel for the sliding architecture would be 917.19 with respect to a mere 19.27 for the nonoverlapping architecture.

In terms of computational cost, the presented architecture is about fifty times less complex than a solution with sliding window, regardless of the computation of *AiF* images.

Another advantage of the proposed architecture is that contrarily to most depth extraction methods, it does not rely on multipliers but on adders only which can drasticaly improve silicon footprint and achievable data throughput.

## Conclusion

In this paper, a hardware architecture for a Depth from Focus algorithm is presented. The algorithm, originally presented in [8] was improved in order to reduce the complexity and memory footprint with a comparable qualitative and quantitative quality. The architecture was designed to operate on a stream of pixel in order to produce All-in-Focus images, depth and confidence information as soon as possible, thus reducing the memory requirements and processing complexity to a minimum. Required complexity for full-HD images amounts to roughly *1.4p* arithmetic operations per depth pixel, with *p*, the number of depth planes, while memory requirements range from 0.06 to 2.06 memory-frames depending on if AiF images are needed. More than 90% reduction of the memory footprint and improvements of two orders of magnitude of the processing power were achieved using a nonoverlapping windows architecture to the price of a limited quality loss.

## References

[1] T. Edeler, K. Ohliger, S. Hussmann and A. Mertins, "Super resolution of time-of-flight depth images under consideration of spatially varying noise variance," 2009 16th IEEE International Conference on Image Processing (ICIP), Cairo, 2009, pp. 1185-1188. doi: 10.1109/ICIP.2009.5413690

[2] X. Gao, R. Kleihorst, P. Meijer and B. Schueler, "Self-rectification and depth estimation of stereo video in a real-time smart camera system," 2008 Second ACM/IEEE International Conference on Distributed Smart Cameras, Stanford, CA, 2008, pp. 1-8.

[3] G. Camellini et al., "3DV — An embedded, dense stereovision-based depth mapping system," 2014 IEEE Intelligent Vehicles Symposium Proceedings, Dearborn, MI, 2014, pp. 1435-1440.

[4] Z. Li et al., "3.7 A 1920×1080 30fps 2.3TOPS/W stereo-depth processor for robust autonomous navigation," 2017 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, 2017, pp. 62-63.

[5] A. Acharyya, D. Hudson, K. W. Chen, T. Feng, C. Y. Kan and T. Nguyen, "Depth estimation from focus and disparity," 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 2016, pp. 3444-3448. doi: 10.1109/ICIP.2016.7532999

[6] Jinbeum Jang, Sangwoo Park, Jieun Jo, and Joonki Paik, "Depth map generation using a single image sensor with phase masks," Opt. Express 24, 12868-12878 (2016)

[7] C. Yeow Wong, S. Liu, S. Chi Liu, Md Arifur Rahman, S. Ching-Feng Lin, G. Jiang, N. Kwok, H. Shi. (2016) "Image contrast enhancement using histogram equalization with maximum intensity coverage". Journal of Modern Optics 63:16, pages 1618-1629.

[8] S. Emberger, L. Alacoque, A. Dupret, J-L. de Bougrenet de la Tocnaye, "Low complexity depth map extraction and all-in-focus rendering for close-to-the-pixel embedded platforms" 2017 ACM International Conference on Distributed Smart Cameras (ICDSC), in press

[9] A. Yokota, T. Yoshida, H. Kashiyama and T. Hamamoto, "High speed depth estimation by smart image sensor system," IEEE International Symposium on Communications and Information Technology, 2004. ISCIT 2004. 2004, pp. 963-968 vol.2. doi: 10.1109/ISCIT.2004.1413862

[10] A. Yokota, T. Yoshida, H. Kashiyama and T. Hamamoto, "High-speed sensing system for depth estimation based on depth-from-focus by using smart imager," 2005 IEEE International Symposium on Circuits and Systems, 2005, pp. 564-567 Vol. 1. doi: 10.1109/ISCAS.2005

[11] J.N.P.Martel, L.K.Müller, S.J.Carey and P.Dudek "High-Speed Depth from Focus on a Programmable Vision Chip Using a Focus Tunable Lens", IEEE International Symposium on Circuits and Systems, ISCAS 2017, Baltimore, pp.1150-1153, May 2017

[12] S. Gehrig, A. Barth, N. Schneider and J. Siegemund, "A multi-cue approach for stereo-based object confidence estimation," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, 2012, pp. 3055-3060.

## Author Biography

*Simon Emberger received the Dipl.-Ing (M.Sc.) degree in electrical engineering from ENSEA School, Cergy, France in 2014 and the M.Sc. in autonomous electrical systems engineering (ISIM-ESA) from Université de Cergy Pontoise, Cergy, France, in 2014. He is currently a PhD student in microelectronic and nanotechnology at CEA-LETI. His current research interest includes embedded processing in CMOS smart sensors for depth map extraction.*