Compact Real-Time Inter-Frame Histogram Builder for 15-Bits High-Speed ToF-Imagers Based on Single-Photon Detection

Ion Vornicu, Angela Darie, Ricardo Carmona-Galán, *Senior Member, IEEE*,
and Ángel Rodríguez-Vázquez, *Fellow, IEEE*

***Abstract*— Time-of-flight (ToF) image sensors based on single-photon detection, i.e., SPADs, require some filtering of pixel readings. Accurate depth measurements are only possible if the jitter of the detector is mitigated. Moreover, the time stamp needs to be effectively separated from uncorrelated noise, such as dark counts and background illumination. A powerful tool for this is building a histogram of a number of pixel readings. Future generation of ToF imagers are seeking to increase spatial and temporal resolution along with the dynamic range and frame rate. Under these circumstances, storing the complete histogram for every pixel becomes practically impossible. Considering that most of the information contained by the histogram represents noise, we propose a highly efficient method to store just the relevant data required for the ToF computation. This method makes use of the shifted inter-frame histogram. It requires a memory as low as 128 times smaller than storing the complete histogram if the pixel values are coded on up to 15 bits. Moreover, a fixed 2^8 words memory is enough to process histograms containing up to 2^{15} bins. In exchange, the overall frame rate only decreases to one half. The hardware implementation of this algorithm is presented. Its remarkable robustness for a low SNR of the ToF estimation is demonstrated by Matlab simulations and FPGA implementation using input data from a SPAD camera prototype.**

***Index Terms*— Shifted inter-frame histogram (SifH), real-time time-of-flight (ToF) estimation, ToF image sensor, single-photon avalanche-diode (SPAD).**

I. Introduction

THE performance of CMOS image sensors based on **S**ingle **P**hoton **A**valanche **D**iodes (SPADs) has been tremendously improved in the last years [1], [2]. They have been proven for photon counting and **T**ime-**o**f-**F**light (ToF) [3]. SPADs are able to work in low illumination conditions with small integration times and to time stamp the arrival of the first detected photon. These features make them suitable for high-speed ToF **C**MOS **I**mage **S**ensors (CIS) [4]. ToF-CIS obtain the depth map of a scene by estimating the ToF at pixel level. Due to the SPAD and **T**ime-**t**o-**D**igital **C**onverter (TDC) ensemble limitations such as uncorrelated noise (e. g. dark counts and background illumination), limited photon detection efficiency, jitter and low illumination conditions, the pixels ToF cannot be estimated from a single measurement. Instead, a relatively large number of measurements is required. From now on, let us call these measurements “inter-frames”, so M inter-frames are required to build the final frame representing an accurate depth image. Even for the best performance SPAD imagers, still several thousands of inter-frames are required [3]. Besides, if the level of the uncorrelated noise is high, then it could trigger the pixels most of the time. For instance, according to the experimental results reported in [5], only 236 detections are true out of $M = 100$k inter-frames. In the remaining 99.76% of the cases, pixels have not been triggered at all or they have been triggered by noise. In these conditions, averaging is not an option. Instead, the computation of the ToF at pixel level involves the finding of the digital code that is repeated most of the time across all acquired inter-frames, i. e. the extraction of the mode. Mode filters have been employed in image processing in the spatial scope [6]. In this occasion, we are going to filter all the time stamps obtained for the same pixel. This problem can be addressed by building ToF histograms at pixel level [7] or by deep learning algorithms [8]. In this paper we only contemplate the first approach. In this way the **S**ignal-to-**N**oise **R**atio (SNR) increases, improving accuracy by $\sqrt{M}$ times. The pixel values (ToF codes) across multiple inter-frames are the addresses of the bins in the histogram memory whilst the content of a memory address represents how many times the address has been encountered (Fig. 1) [5]. When it comes to real-time ToF computation at ultra-high speed (hundreds of thousands of inter-frames per second) for wide range (hundreds of nanoseconds) with few picoseconds temporal resolution and large spatial resolution (kpixel-array, 1 kpixel = 1024 pixels), the memory size required to store the histogram becomes too large and full random access time requirements become critical. In order to have a better

Manuscript received October 24, 2018; accepted November 23, 2018. Date of publication December 10, 2018; date of current version February 15, 2019. This work was supported in part by the Office of Naval Research (ONR), USA, under Grant N000141410355, in part by the Spanish MINECO and the European Region Development Fund (ERDF/FEDER) through the iCaveats Project under Grant TEC2015-66878-C3-1-R, in part by the Junta de Andalucía through the SmartCIS3D Project under Grant TIC 2338-2013, and in part by EU-REA through the Achieve Project (EU H2020 MSCA-ITN 2017) under Grant No. 765866. The associate editor coordinating the review of this paper and approving it for publication was Prof. Diego Barrettino. *(Corresponding author: Ion Vornicu.)*

The authors are with the Instituto de Microelectrónica de Sevilla (IMSE-CNM), CSIC, Universidad de Sevilla, 41092 Sevilla, Spain (e-mail: ivornicu@imse-cnm.csic.es).

Digital Object Identifier 10.1109/JSEN.2018.2885960

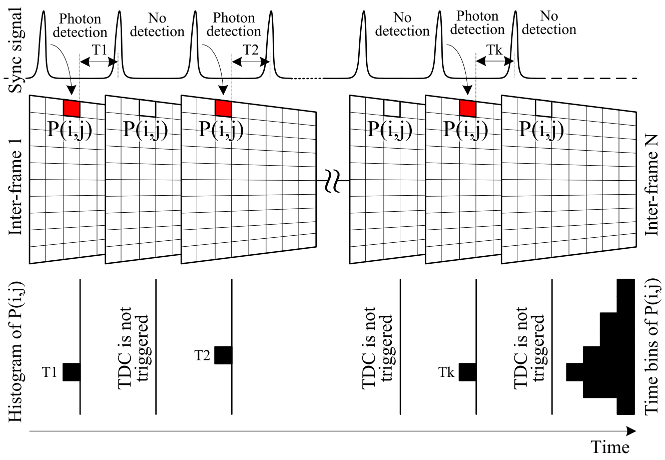

Fig. 1. Block diagram of inter-frame histogram building.

understanding of the challenges designing a pixel level inter-frame histogram builder and the limitations of histograms storage, let us give some numbers:

i) Suppose that ToFs estimated by each pixel are coded on 15 bits, the complete histogram for one single pixel has 32 kbins (1 kbin = 1024 bins). If each of the bins of the histogram is coded on 10 bits, i.e. $N_h = 10$, the pixel histogram requires 320 kb of memory. For an array of 64 × 64 pixels, the memory footprint of the complete histogram will be 1.25 Gb.

ii) Concerning the access to the histogram memory, let us consider a chip throughput of 1.6 Gbps. This assumption takes in consideration the switching performance of the digital output pads (50 MHz) and the level of parallelization limited by the power ring budget and package number of pins (32 channels). As each pixel value is coded on 15 bits, full random read and write access times have to be less than 9.4 ns. No DDR memory off-the-shelf meets these specifications because they have been designed to be faster in burst mode and they have large Read/ Write, Active, Precharge and Refresh latencies. Even though DDR technology gets faster, the memory module also became larger such that the latency stays the same. This limitation naturally calls for parallelism. If the total memory is divided in 32 channels, then the access timing constraint is relaxed to 300 ns. On top of this, the memory shrinkage associated to the division in channels implies a smaller latency. However the memory size still remains the biggest issue, e. g. it is too large to fit in a FPGA's Block RAM (BRAM). Moreover, an ASIC implementation of an SRAM memory of 40 Mb per single channel of 2 × 64-pixels still requires an area (more than 40 mm^2 in a 90nm CMOS process) that is too large to be affordable. For these reasons it is not possible to store the complete histogram for every pixel.

Seeking to decrease the histogram memory size, the following algorithms are considered: **P**artitioned **i**nter-**f**rame **H**istogram (PifH) and **F**olded **i**nter-**f**rame **H**istogram (FifH).

PifH is storing only a part of the complete histogram at a time. This approach is referred as time gated scanning technique [9]. Consider a partial histogram of only 2^{N_b} bins, where N_b is the number of bits of the partial histogram memory. If the number of bits per pixel is N_p, this algorithm will require to build $2^{N_p-N_b}$ partial histograms. For instance, pixel values on 15 bits can be represented in a histogram with $2^{15} = 32768$ bins. In order to overcome the border effect, the partial histograms have to overlap. If each partial histogram contains, for instance, only 2^8 bins, this algorithm will require building at least 128 partial histograms. As the data of the partial histograms can be discarded after processing, the histogram memory in this case is at least $2^{N_p-N_b}$ times smaller (i.e. 128 times in the example). After scanning the entire dynamic range, one last partial histogram might be required around the peak detected in the early ToF estimation phase. It ensures that the ToF information is not truncated between consecutive partitions. However, the overall frame rate is also decreased by the total number of required partial histograms. For this reason PifH is more appropriate for moderate values of N_p (up to 10 bits).

FifH algorithm consists of building partial histograms by clustering the pixel value without overlapping [10]. Let us suppose two clusters: one corresponding to the least significant N_b bits of the pixel value coded on N_p bits; the other corresponding to the $\left(N_p - N_b\right)$ most significant bits. In addition to requiring the same memory footprint as PifH, FifH has an overall frame rate only 2 times smaller comparing to the approach that stores the **C**omplete **i**nter-**f**rame **H**istogram (CifH). However, even if this technique is suitable for hardware implementation, it requires additional compensation for the uncertainty errors that occurs when the ToF Gaussian bell is centered at multiples of 2^{N_b} bins. Moreover, it is worth to mention that the SNR of both histograms is affected by noise folding.

This work presents a novel approach to efficiently store the **i**nter-**f**rame **H**istograms (ifH) without losing the accuracy of the ToF estimation. The basic idea of the proposed **S**hifted **i**nter-**f**rame **H**istogram (SifH) algorithm relies on the following observations: the uncorrelated noise is uniformly distributed on the histogram's floor and the ToF information is concentrated in the Gaussian bell. Therefore storing the CifH is not necessary. Instead, only 2^{N_b} bins centered on the ToF data are enough to be stored. In order to do that, all the time stamps have to be shifted to the N_b-bit base address band. Consequently, the required memory is much smaller and, above all, fixed while N_p can vary over a range of values, e. g. from 8 to 15 bits. This is an extraordinary advantage of this algorithm because it allows to dynamically change N_p and to maximize the frame rate depending on the dynamic range and temporal resolution requirements.

Ultra-high speed ToF sensors demand real-time ToF computation. We propose a circuit to realize such estimate on-the-fly while the pixel ifH is collected. It is based on the detection of the ifH peak, i. e. a mode filter operating on all the ToF measurements acquired by each pixel. Extraction of the mode is rather preferred than the histogram center of mass because it can be implemented with a simpler hardware that requires a smaller memory footprint. Notice that this choice relies on the assumption that the histogram has a Gaussian shape. In these conditions, the depth image is ready as soon as the acquisition of the inter-frames ends, with a latency of just one inter-frame acquisition.

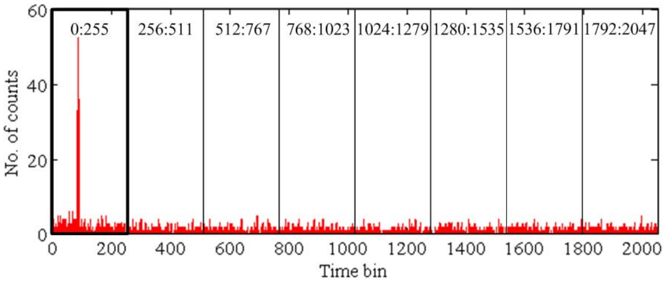

Fig. 2. Histogram partitioning.

The paper is organized as follows: Section II presents the PifH algorithm as a solution to decrease to histogram memory requirements. Section III focuses on the FifH algorithm, which represents an advance over PifH due to its suitability to operate on the outputs of high-speed and high resolution imagers. The limitations of these algorithms are discussed as well. Section IV concentrates on the novel SifH algorithm, which overcomes all the limitation in terms of frame-rate, area, and N_p. The key parameters of the algorithm are computed. The reliability and robustness of the algorithm are confirmed by Matlab simulations and experimental results obtained with a full custom FPGA implementation using input data provided by a SPAD camera prototype. Section V presents the proposed hardware for the SifH algorithm. It has been implemented on a Spartan3 FPGA with very low resources. Section VI contemplates the scalability of the design for large arrays in the case of ASIC and FPGAs implementations. Section VII extrapolates the implementation of the peak detector for a SifH channel incorporating 128 pixels. Section VIII is dedicated to conclusions.

II. Partitioned Inter-Frame Histograms (PifH)

This is a quite straightforward implementation, requiring only to compare the pixel values to a threshold corresponding to the extremes of each partial histogram —see Fig. 2 in which $N_b = 8$. If each pixel value is coded on $N_p = 11$, then there are $2^{N_p - N_b} = 8$ partitions. Considering again that $N_h = 10$ —which is a practical value derived from the fraction of events that correspond to a true measurement in practice—, each partition requires $N_h \times 2^{N_b} = 2.5$ kb of physical memory, which means 8 times less memory than CifH. Seamless scalability for larger N_p is the major advantage of this approach. This can be achieved by using the same memory footprint per partial histogram and multiplexing it in time. The major disadvantage is that the complete histogram has to be scanned until the true ToF data are found. For better accuracy and ToF information integrity, the partitions have to overlap. Moreover, after scanning and peak detection, one last histogram has to be acquired centered on the peak detected in the scanning phase. This means that the overall frame rate decreases at least by $2^{N_p - N_b}$ times, depending on the overlapping ratio.

The accurate ToF measurement is the mode of the CifH. It will be denoted by B_M as it is the position of the bin rendering the largest value. The position of the bin rendering the largest value of a partition is b_M. If P_M is the index of the partition containing the global maximum and the partitions are not overlapping, B_M can be computed as:

$$B_M = 2^{N_b} (P_M - 1) + b_M \tag{1}$$

In order to improve the overall ToF computation rate, the partitions during scanning can have less samples [9], in which case, the uncertainty error might increase. If speed is not a concern, then this method can be successfully applied.

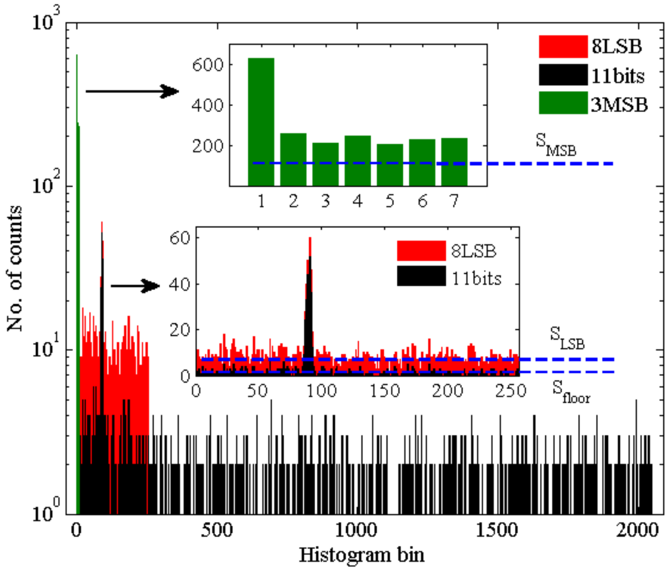

Fig. 3. Histogram folding.

III. Folded Inter-Frame Histograms (FifH)

This approach is based on applying masks on the incoming time stamps in order to build 2 different histograms as follows: the first one built from the most significant $(N_p - N_b)$ bits and the second one built from the N_b less significant bits. We have used CifH data provided by the SPAD-CAM prototype [11] as input data in order to illustrate how FifH technique works (see Fig. 3 – 11 bits marker). Compared to CifH, the memory footprint is decreased by $2^{N_p - N_b} = 8$ times. Another important observation is related to the noise floor of the **M**ost **S**ignificant **B**it (MSB) histogram, S_{MSB}, which is larger than the one of the **L**east **S**ignificant **B**it (LSB) histogram, S_{LSB} which in turn is larger than the noise of the CifH on 2^{N_p} bins, S_{floor} (Fig. 3 – upper and lower insets). This happens because the noise that is spread along the CifH folds into the MSB and LSB histograms. Obviously the smaller the number of bins of representation, the higher the folding order.

The ToF measurement, B_M after performing the 2-step acquisition is computed as:

$$B_M = 2^{N_b} \left(b_{M,MSB} - 1\right) + b_{M,LSB} \tag{2}$$

where $b_{M,MSB}$ and $b_{M,LSB}$ represent the position of the maximum values in the MSB and LSB histograms. They are used to compute a first approximation of the ToF.

The major advantage of this algorithm is that it can be employed with pixels values represented with up to 15 bits by using the same amount of physical memory as in the PifH. This technique requires only 2 acquisitions, no matter N_p.

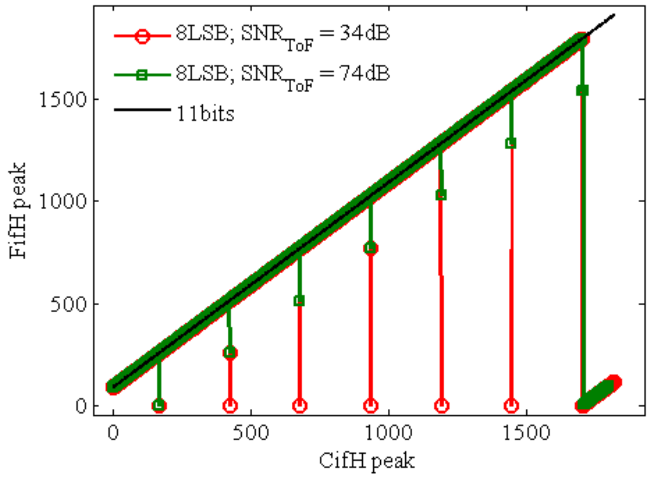

Fig. 4. FifH uncertainty error.

This is another important advantage compared to PifH where the overall frame rate decreases at least by $2^{N_p - N_b}$.

The major drawback of this technique is the uncertainty error that occurs whenever the Gaussian bell is swept through multiples of 2^{N_b}. In this case the peak of the histogram is misplaced (see Fig. 4 – circle marker). Note that the border error is larger than 255 bins. This happens due to inherent noise folding effect of the FifH technique. This is why a MSB histogram has a lower SNR which makes it prone to detect false peaks. The abnormal border errors correspond to 3 bits-MSB (3MSB) histogram having the peak on the first bin. The error could be lowered by increasing the SNR$_{\mathrm{ToF}}$ of the input data (Fig. 4 – square marker). In this case, the 3MSB histogram detects the peak with a boarder error around 255 bins, rather than detecting false positives only on the first bin.

Further corrections of the uncertainty points are required by acquiring **F**ine **H**istograms (FH) centered on B_M. These histograms must have at least 2^{N_b+1} bins, i.e. 512 with this occasion. Even so, when located at the FH borders, the ToF information could be truncated. This is not acceptable when ToF is required to be computed more accurately such as the center of mass of the ToF information. Therefore, additional bins are required as safety margin for the FH. After all, it looks like the footprint of the FH exceeds the 8 bits address space allocated for the MSB and LSB histograms.

FifH technique to compute the first approximation of the ToF has been recently reported as **P**artial **H**istogram **R**eadout (PHR) [10]. PHR employs 3 **C**oarse **H**istograms (CH) instead of 2 (previously called MSB and LSB histograms). These 3 CHs coded on 3 bits are successively accumulated for coarse estimation of 10 bits-ToF. Similar to eq. (2), the coarse approximation of the ToF is computed as $B_M = 2^7 b_{M,97} + 2^4 b_{M,64} + 2^1 b_{M,31}$, where $b_{M,31}$, $b_{M,64}$ and $b_{M,97}$ are the position of the peak in the CHs built from the [3:1], [6:4] and [9:7] bits out of the [9:0] bits of the pixel values.

One particularity of FifH technique is that the bits corresponding to the CHs do not overlap. Consequently, as predicted by the FifH approach, PHR is prone to uncertainty errors. This is proved by the simulation results presented in Fig. 5. It shows a parametric simulation by sweeping the peak of 10 bits-CifH along the entire dynamic range with 1 bin step. The true peak of CifH is compared to the coarse approximation computed by PHR (CH-PHR) and CH-6MSB which is built by the bits [9:4] of the pixel value. Thus, CH-PHR suffers of border errors of 2^7 or 2^4 or 2^1 bins (Fig. 5-red curve). This means that FH would require at least 2^8 bins to encompass the ToF information. It is worth to mention that the number of border errors becomes even larger for smaller SNR$_{\mathrm{ToF}}$. This is due to the noise folding effect which implies a higher noise floor for a smaller CH (see Fig. 3).

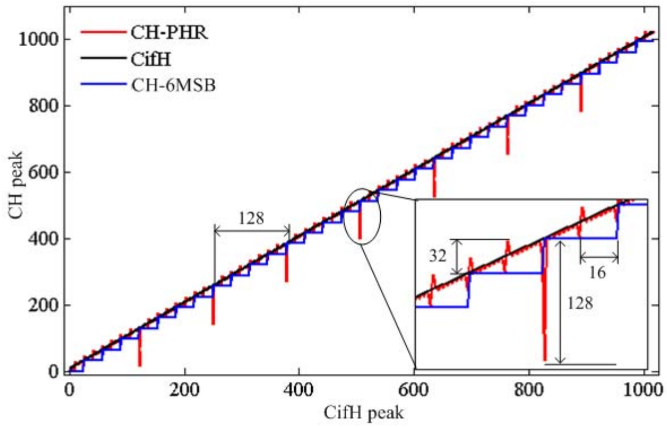

Fig. 5. Coarse peak estimation by CH-PHR and CH-6MSB. True peak of CifH with SNR$_{\mathrm{ToF}}$ of 34dB.

Although CH-6MSB makes a coarser approximation than CH-PHR, it is more accurate because does not exhibit border errors. Therefore CH-6MSB eventually requires smaller FH then CH-PHR does. For this reason this approach is contemplated in the next section related to the proposed SifH algorithm. Obviously CH-PHR occupies less memory then CH-6MSB but also involves much larger FHs to resolve the border errors. Moreover, the coarse approximation of the peak by CH-PHR is 3× slower than CH-6MSB.

Thus, multiple non-overlapping CHs are not suitable for larger ToF depths due to lower computation rate and larger uncertainty errors. For this reason the proposed SifH algorithm is based on a single MSB histogram. Its size is optimized for computation speed, memory footprint and accuracy for different ToF depths up to 15 bits. SifH is extensively presented in Section IV. A comparison with the PHR approach will be presented as well.

IV. Shifted Inter-Frame Histograms (SifH)

SifH completely eliminates the uncertainty error of the FifH approach which involves large FHs for linearity corrections. Besides, the physical memory requirements and overall frame rate remain the same. Thus, by only using $N_h \times 2^{N_b}$-bits SRAM memory, the ToF can be accurately computed on-the-fly while N_p can vary in a range that goes up to 15 bits.

This feature is very important for the next generation of 3D cameras that will require simultaneously both, picosecond time resolution and a wide dynamic range. Under these circumstances, the representation of pixel values by 15 bits codes could be quite common. It means that the CifH for a single pixel would have 32768 bins. In this case, the required physical memory is of 320 kb, leading to an outrageous amount of 1.25 Gb for a 64 × 64-pixel array. Obviously, building the complete histogram is not a good solution anymore [11], [13]. As mentioned in Section III, PHR algorithm based on the FifH approach is a good option for ToF coded on lower number of bits. However it is prone to border errors, as it will be demonstrated later on by comparing the ToF computation based on SifH and PHR.

The proposed SifH algorithm achieves a remarkable memory reduction, down to 128 times smaller than CifH of 15 bits-ToF. It is highly accurate over the entire dynamic range even in low SNR_{ToF}. To the best of our knowledge, these specifications are reported for the first time. Moreover, SifH is able to adapt to different N_p. Thus, frame rate can increase when accuracy or dynamic range are less demanding.

Before explaining the principle of SifH, it is worth to mention that a per-pixel histogram always contains 2 key parts: the Gaussian bell which encodes the ToF and SPAD-TDC jitter; and the noise floor which encodes the amount of uncorrelated noise in the pixel. Let us suppose a typical ToF CifH retrieved from the SPAD-CAM [11] (see Fig. 2). The specifications of the SPAD imager along with the experimental setup are presented in [5]: i) the average **D**ark **C**ount **R**ate (DCR) at 1V and the **P**hoton **D**etection **E**fficiency (PDE) at 640nm are 42 kHz and 5%; ii) the imager operates in gating mode with 300ns time gate; iii) the irradiance is below 10nW/mm^2; iv) each 11 bits-CifH is built out of 65536 inter-frames. Analyzing the ToF histogram, one can realize that it is not necessary to store the entire noise floor by building the CifH because the majority of the bins contain redundant information of the pixel noise. In fact, only using a reduced amount of bins is enough to accurately compute the ToF. Note that the histogram spans over about 300ns. Moreover the histogram accumulates the noise by measuring the time interval from the first occurrence of a noise pulse, after the time gate opens, to a synchronization pulse coming from the laser. This explains the uniformly distributed noise shape.

The key is to find a method to virtually zoom into the CifH such that the Gaussian bell is captured by a smaller histogram of just 2^{N_b} bins. The SifH algorithm consists in building 2 histograms on a N_b bits address —for illustration purposes $N_b = 8$ for $N_p = 15, 14$ and $N_b = 6$ for $N_p = 11, 10$:

- The first one is used to compute a coarse approximation of the ToF data by extracting the position of the peak in the histogram, $b_{M,coarse}$. The CH is built from the incoming pixel values previously filtered by applying a mask on the N_b-MSB (Fig. 6 – black/red curves: CifHs are on 15/11 bits). Note that the 11 bits-CifH has been expanded to 15 bits-CifH as follows: the ToF peak has been separated from the noise floor; the noise floor of the 15 bits-CifH is built by concatenating the noise floor of the 11 bits-CifH. The choice of this coarse estimation approach has been discussed in Section III by comparing it with the CH-PHR approach. As mentioned before, due to noise folding effect, each additional 20dB of SNR-CifH entails an increase of SNR-CH by only 8dB.

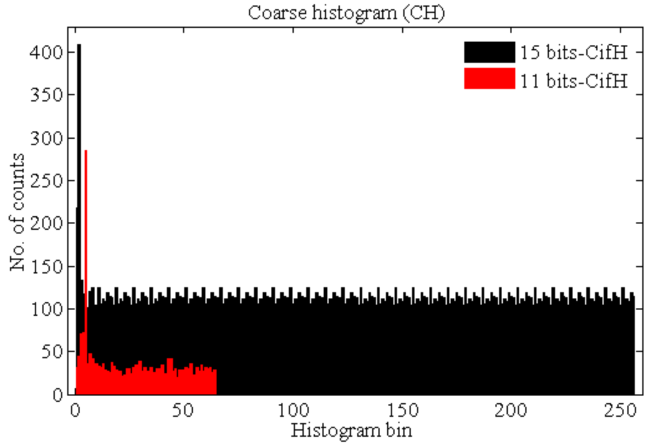

Fig. 6. First histogram (CH) on 256 bins (black color) and 64 bins (red color) when CifH has 32768 bins and 2048 bins.

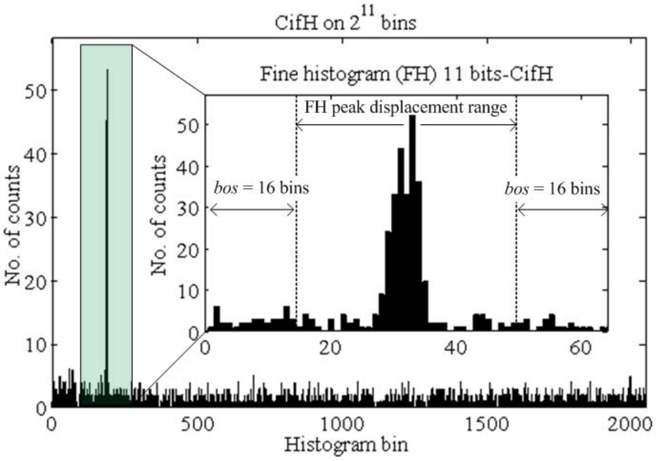

Fig. 7. Second histogram (FH) on 64 bins and CifH on 2048 bins.

- The second one is centered on the ToF data such that we keep the same accuracy of the CifH no matter N_p (Fig. 7). It is achieved by shifting the pixel values, pix_val such that they can be mapped on a histogram of 2^{N_b} bins.

Unlike FifH and PHR, SifH employs the same number of bits for both CH and FH. It is computed as $N_b = \lfloor N_p/2 \rfloor + 1$. This means that at least one bit overlapping occurs between CH and FH. For instance if $N_p = 15$ then $N_b = 8$. Note that for $N_p = 14$, N_b has the same value as for 15 bits. For even N_p, two bits overlap which is even better from the SNR_{ToF} point of view.

The thresholds of the filter are computed as:

$$TH_+ = 2^{N_p - N_b} b_{M,coarse} + SB - b_{os} \quad (3)$$

$$TH_- = 2^{N_p - N_b} b_{M,coarse} - SB - b_{os} \quad (4)$$

$$SB = 2^{N_b - 1}; \quad b_{os} = 2^{N_b}/4 \quad (5)$$

where SB is the number of side-band bins, and b_{os} is the offset eventually needed to correct the position of the peak as follows. If N_p is of 15 bits then N_b has to be of 8 bits. This means that FH peak can be located anywhere along $2^{N_p - N_b}$ bins. At both ends, the ToF information might be truncated, especially for large Gaussian FWHM. The solution is to use a larger number of bins (256 in this case).

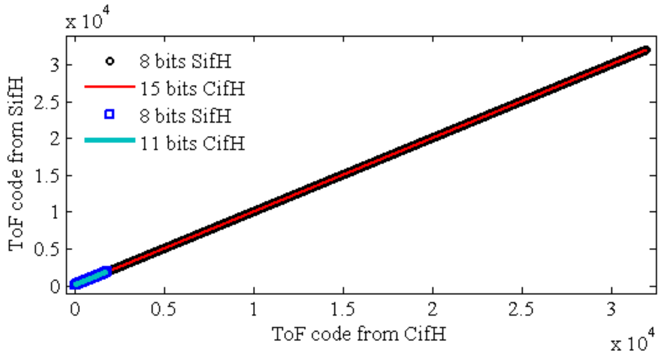

Fig. 8. Comparison between ToF code computed from 15 and 11 bits CifH and SifH on 8 bits.

The b_{os} parameter helps to center the FH displacement range on the FH addresses space (see Fig. 7-inset).

The shifting value, Δ to be able to map the filtered pixel values on N_b-bit histogram is computed as follow:

$$\Delta = \left[\text{floor}\left(\frac{2^{N_p-N_b} b_{M,coarse} + SB - b_{os}}{2^{N_b}}\right) - 1\right] \cdot 2^{N_b} + \text{mod}\left(\frac{2^{N_p-N_b} b_{M,coarse} + SB - b_{os}}{2^{N_b}}\right) \quad (6)$$

where "floor" and "mod" are the quotient and rest of the division by 2^{N_b}.

At this point we have to map on N_b-bit histogram the incoming pixel values which have passed through the filter such that:

$$TH_+ \geq pix\_val > TH_- \quad (7)$$

Note that $pix\_val - \Delta < 2^{N_b}$ when the CifH has 2^{N_p} bins, i.e. $pixel\_val < 2^{N_p}$.

After the FH is built, the precise position of ToF data is obtained by extracting the position of the peak in the histogram, $b_{M,fine}$. Finally, the accurate ToF code is computed as:

$$B_M = b_{M,fine} + \Delta \quad (8)$$

where B_M is actually the pix_val that was repeated most of the time along the acquisition phase.

It is worth to mention that even though Δ could be affected by error when the ToF data is positioned at multiples of 2^{N_b}, it does not affect the B_M. The reason is that Δ is subtracted from the pixel values to build the FH and subsequently added to compute the accurate ToF, such that an auto-zero compensation is automatically performed. Thus SifH is uncertainty error free even if the SNR_{ToF} of the CifH is as small as 34dB. This is proved by Fig. 8 where the histogram peak is swept across the full dynamic range of the CifH. The continuous lines represent the histogram peak extracted from the CifH. The square and circle markers represent the histograms peaks computed by the SifH algorithm using only 8 bits histograms. There is a perfect match between Matlab simulations and experimental results. Note that the input data used in Matlab simulations have been fed to the FPGA implementation through a pattern generator.

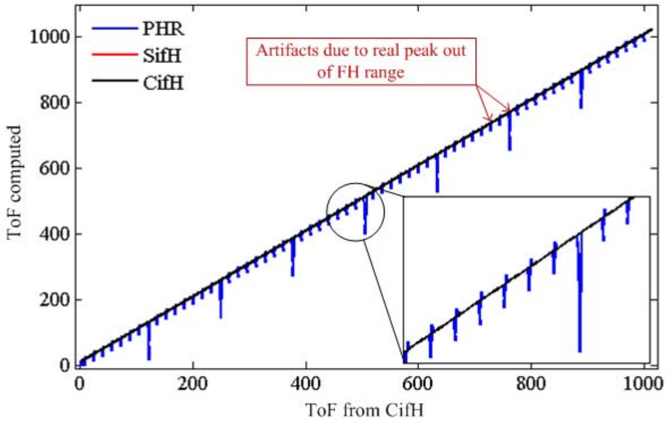

Fig. 9. ToF computed by SifH, PHR and CifH; $\text{SNR}_{\text{ToF}} = 34$dB.

In order to have a fair comparison with the PHR algorithm, SifH has been down scaled to operate 10 bits-CifH. As explained in Section III, PHR coarse peak could jump by ±16 bins or ±128 bins. The FH of the PHR has ±8 bins around the coarse peak. FH is always centered on the coarse peak. This means that, if the coarse peak is deviated by 16 bins, the fine peak would be out of the FH range. Even with 32 bins, the fine peak could be located at one of the FH's ends where peak detection is not safe to operate. Consequently, PHR fails to resolve the ToF even for $2^4\times$ border errors, as Fig. 9 demonstrates. The $2^4\times$ and $2^7\times$ border errors (Fig. 9-blue curve) might not be seen in the distance ranging experiment because it has too less points [10]. As predicted, these errors have been solved by employing a FH of 34 bins, instead of 16 bins around the coarse estimation. However, the larger errors cannot be resolved by 34 bins-FHs. It requires at least 2^{7+1} bins. Therefore the apparent advantage of smaller CH-PHR footprint compared to CH-SifH footprint is cancelled.

Instead, FH-SifH on 64 bins (6 bits) has a considerable margin to operate error free even if the FWHM of the ToF data is large. Moreover, the b_{os} parameter can adjust the FH address space such as the fine peak is never truncated. This is the key difference that allows SifH to compute ToF error free, along the entire dynamic range of the sensor (Fig. 9-red curve). Moreover, SifH keeps working perfectly at 24dB, equivalent to a high level of uncorrelated noise of 20× DCR. In this case, PHR increases the number of ambiguity points.

Unlike SifH, PHR cannot be scaled up because the ToF computation rate gets even lower than SifH (e.g. 5× slower for $N_p = 15$ bits). Moreover, FH-PHR memory requirement exceed by far the one for CH-PHR (e.g. 2^{13+1} bins for $N_p = 15$ bits). Therefore SifH compared to PHR is faster and more accurate. SifH occupies less memory for larger depth ToF.

V. SifH Building Blocks

In this implementation, we will be considering that N_p is 15 bits, N_h is 10 and N_b is equal to 8. The design of the real-time SifH has been implemented on a Spartan3 FPGA. It includes the following main blocks (see Fig. 10): a N_p-bit serial-input parallel-output (SIPO), a N_p-bit parallel-input parallel-output register (PIPO), one digital filter (DF),

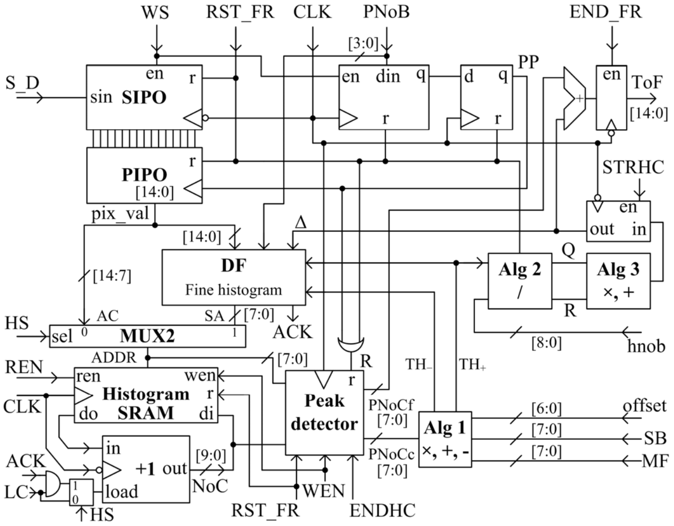

Fig. 10. Block diagram of SifH.

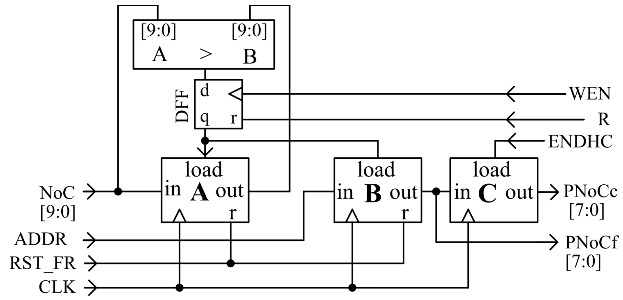

Fig. 11. Circuit diagram of peak detector.

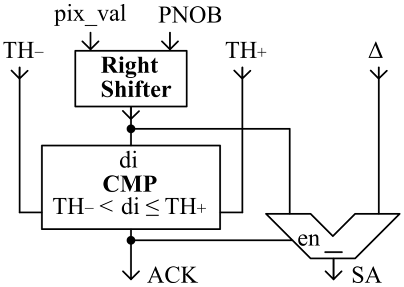

Fig. 12. Circuit diagram of DF.

a multiplexer with 2 inputs on N_p bits, a $N_h \times 2^{N_b}$-bit SRAM memory, a N_h-bit register with one step automatic increment, a peak detector circuit, 3 algebraic circuits to compute additions, subtractions, multiplications and divisions (Alg1, 2 and 3) and $2 \times N_p$-bit memory to store the Δ value and accurate ToF of the pixel.

The hardware implementation of the SifH algorithm of a single pixel is operating as follows:

i) The global reset (RST_FR) is activated before starting to capture a new frame. At this point it is important to reset the *Histogram SRAM* and the *Peak detector's* A and B registers. Later on, the pixel value is serially loaded from the SPAD imager through the SIPO, starting with LSB, on the negative edge of the clock (CLK), while the signal WS is set low (shift enabled). N_p is defined by the signal PNoB. Every N_p periods of CLK the content of SIPO is loaded into PIPO, on the positive edge of the signal PP. Therefore the output of the PIPO, called pix_val, is stable $N_p \times \mathrm{T_{CLK}}$ time period while it has to be placed in the right bin of the histogram.

ii) The first step is to build the coarse histogram by setting the signal HS low.

When REN is set high, the memory located at the address indicated by the pixel value is read out on the positive edge of CLK. It represents how many times the current pixel value has been encountered before. It is incremented by 1 unit when LC is set high, on the negative edge of CLK. Subsequently, the updated **N**umber **o**f **C**ounts (NoC) is overwritten in the histogram SRAM at the same address, when WEN is set high, on the positive edge of CLK. Next, if the current NoC is bigger than its previous value, it is overwritten along with the corresponding address ADDR in the *Peak detector*'s registers, A and B respectively (see Fig. 11).

The acquisition of the coarse histogram ends right after the M-th pixel value is resolved. When ENDHC is set high, the pixel value that has been encountered most of the times in the CH is stored by register C to compute TH_-, TH_+ and Δ. It is worth to mention that the division operation has been implemented by a sequential scheme due to area constraints. Thus the operation is completed in $N_p \times \mathrm{T_{CLK}}$. Under these circumstances, it is better to store the Δ value and recall it whenever is needed. This is the purpose of the register enabled by the signal STRHC.

Unlike Δ, TH_- and TH_+ are not required to be stored because they are computed quite fast from the position of the final peak of the CH, PNoCc.

iii) At this point the *Histogram SRAM* and the temporal registers A and B of the *Peak detector* have to be reset again. The second acquisition can start to build the FH. The signal HS is set high such that the pixel value is routed through *DF* block (see Fig. 12). Note that the construction of the FH and the detection of the peak value are the same as in the previous step. The only differences are in the filtering of the input pixel values and the condition to count them in the histogram bins. First of all, some data alignment is required depending on the input PNoB. This is required only for adaptive frame rate applications where PNoB can vary from frame to frame. Subsequently, each incoming pixel value is checked whether it is within the limits computed at the previous step. If so, then the pixel value translated on N_b-bit histogram address is placed in the corresponding bin when both signals ACK and LC are high. The translation is required to fit a N_p-bit pixel value on a N_b-bit histogram address. It is done by merely subtracting the Δ value computed in the previous step from the incoming pixel value. The final peak value address of the fine histogram, PNoCf is available right after the acquisition of the FH ends by placing the last pixel value in the right bin.

Finally, the accurate ToF is computed on chip by adding the Δ value to the peak position of the FH, $b_{M,fine}$. The storage of the ToF is enabled by the signal END_FR, on the negative edge of CLK. The signals chronogram is depicted in Fig. 13. HC-Pi and HC-Pf labels stand for the initial and final peak values of the CH corresponding to a certain pixel. HF-Pi label represents the initial peak value of the FH corresponding to the same pixel.

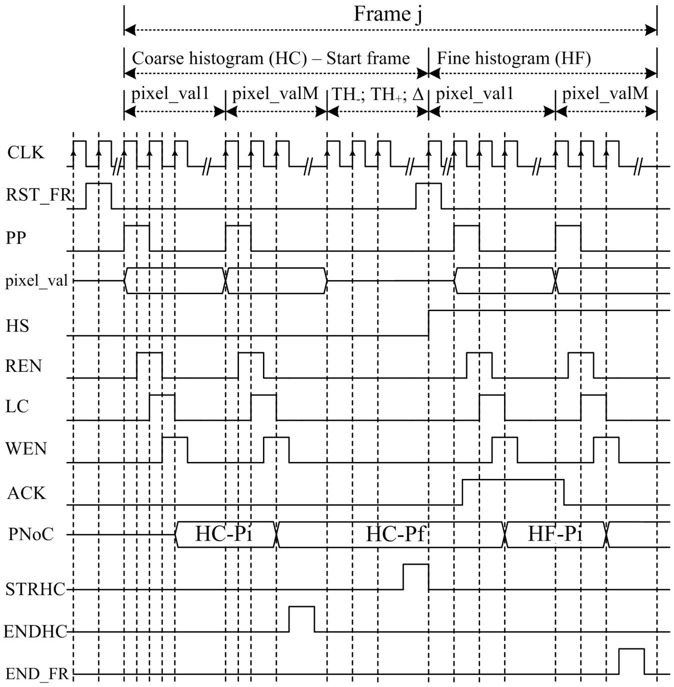

Fig. 13. Signals chronogram.

Thus, instead of sending to USB up to 320 kb of histogram per pixel to estimate the ToF off-chip, we send directly only the accurate ToF represented on maximum N_p bits.

VI. SifH Scalability

The proposed SifH algorithm can be easily scaled to any pixel-array size and it is suitable for ultra-high frame rate imagers. Under these circumstances, in order to relax the time constraints on the readout and bin placement circuits, parallelism is naturally called.

Let us make the calculations for a 64×64-pixels ToF image sensor with 32 outputs at 50 MHz with an equivalent throughput of 1.6 Gbps and an inter-frame rate of 26 kfps to 48 kfps when PNoB changes from 15 to 8 bits.

The time constraint on the histogram memory access is given by the minimum PNoB. Suppose that each serial output is connected to a SifH channel which has to resolve 128 pixels. This means that the memory resources from Fig. 10 have to be multiplied by 128, i.e. the 2.5 kbits histogram memory and 56 bits register allocated as follows: 2×8 bits to store the address of the peak values of the coarse/fine histograms, 10 bits for the peak value and 2×15 bits for Δ value and ToF.

For an ASIC implementation in a 90nm CMOS process, the histogram memory footprint for one channel is below 0.4 mm^2. It easily fits into a mini ASIC with an affordable price. In the end, 32 channels could be encapsulated in the same package.

SifH algorithm is also suitable to be integrated on-chip with the imager if it is shared by multiple pixels.

For FPGA implementation, one has to consider the total histogram memory requirement from 8 to 16 Mb depending whether the number of counts per bin is represented on 8 to 16 bits. However some Xilinx FPGA, such as XC7K160T (325×36 kbits BRAM), XC7A200T (365×36 kbits BRAM) or XC7K355T (715×36 kbits BRAM) [14] could accommodate all 32 SifH channels.

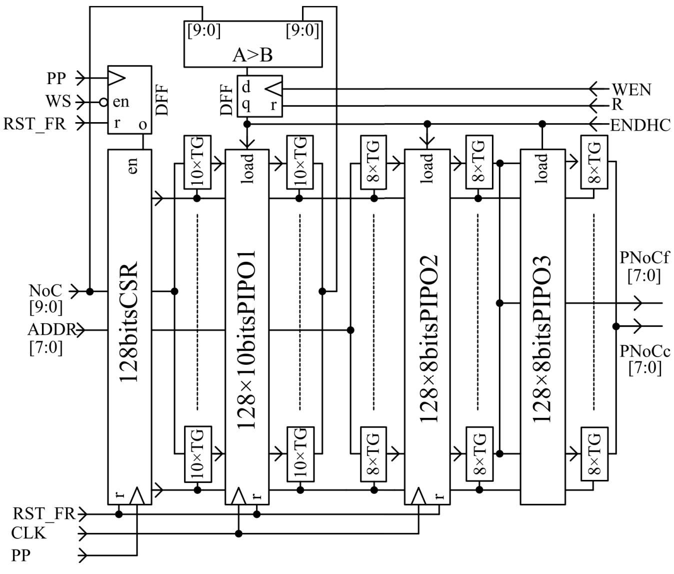

Fig. 14. Single channel of real-time peak detector.

For the sake of simplicity the next section presents only the scaling of the peak detector circuit. The multiplexing scheme is similar for the Δ and ToF registers.

VII. On-Chip ToF Computation

Ultra-high inter-frame rate 3D SPAD imagers require the computation of ToF on-the-fly. Consequently it has to be implemented on-chip and parallelized over different channels. For this purpose we propose a hardware to compute the peak of each histogram in real-time. The design for one pixel (Fig. 11) is scaled for one channel of 128 pixels. The block diagram of one out of 32 channels is presented in Fig. 14.

It is based on the fact that each bin is accessed during the histogram building phase. Thus the peak value is updated each time a new bin value is read from the histogram memory. Thus the histogram peak is available right after the acquisition phase ends.

It is built by a pixel decoder, a 128×10 bits PIPO register to store the histograms peak, a 128×8 bits PIPO to store the peak address and a 128×8 bits PIPO to keep the final peak address of the CH which is used to compute the corresponding TH_+ and TH_-. Note that PIPO register can be replaced by latches.

The pixel decoder is implemented by a 128 bits circular shift register (CSR) and transmission gates (TG). It also can be used to multiplex the registers that store the Δ value and ToF of each pixel.

The circuit is operating as follows: PIPO 1, 2 are reset at the beginning of each frame. Before receiving the first inter-frame, the CSR is reset as well, selecting the first position in PIPO 1-3, allocated for the first pixel in the channel.

The maximum NoC in the histogram of the first pixel and the corresponding ADDR are updated on the positive edge of WEN signal. On the next edge of the signal PP, the shift register selects the second position in PIPO 1-3, allocated for the second pixel and so on up to the last pixel in the channel of the first inter-frame.

After the last inter-frame is processed the final peak addresses in the pixels' CH are stored in PIPO 3 by activating the signal ENDHC. At the end of the FH acquisition, the peaks addresses of the FHs are stored by PIPO 2.

VIII. Conclusion

This paper concentrates on noise reduction in ToF SPAD imagers based on pixel level inter-frame histogram building. We have discussed the area limitations of storing the CifH when N_p is up to 15 bits. Two different alternatives have been contemplated to reduce the histogram memory requirements, i.e. PifH and FifH. However the frame rate of the former technique is strongly affected by N_p. The latter technique requires some additional correction of the uncertainly errors and it is affected by noise folding.

In order to overcome the aforementioned limitations, we propose a method to efficiently store histograms in real-time. SifH is highly efficient and suitable for kpixels high speed imagers and large ToF depths. An extensive comparison of the SifH algorithm to a recently reported PHR algorithm based on FifH has been presented as well. SifH has the following advantages: *i*) it requires very low memory footprint (2.5 kbits or 256 bins/ ifH); *ii)* the memory footprint is fixed for N_p of 8 bits up to 15 bits; *iii)* the required memory footprint for the same N_p tremendously decreases up to 128 times; *iv)* the accuracy of peak detection is not affected by increasing N_p even if the ifH number of bins is fixed; *v*) SifH is faster than PHR and the computation speed does not depend on N_p; *vi)* unlike PHR, SifH is free of uncertainty errors along the entire dynamic range; *vii)* SifH can be scaled up to 15 bits-ToF. In order to cancel the border errors, PHR would need larger FH than FH-SifH.

The SifH hardware implementation for one pixel is thoroughly presented. The scalability towards ultra-high frame rate ToF imagers is discussed. All the calculations are taking into account for 64 × 64-pixels array.

We have also proposed a custom design to extract in real-time the peak of the pixel ifHs of a channel incorporating 128 pixels on up to 15 bits. The integration of the proposed algorithm in ASIC and FPGAs is addressed as well.

References

[1] G. F. Dalla Betta, L. Pancheri, D. Stoppa, R. Henderson, and J. Richardson, "Avalanche photodiodes in submicron CMOS technologies for high-sensitivity imaging," in *Advances in Photodiodes*. Rijeka, Croatia: InTech, 2011, pp. 225–248, [Online]. Available: http://www.intechopen.com/books/advances-in-photodiodes/avalanche-photodiodes-in-submicron-cmos-technologies-for-high-sensitivity-imaging doi: 10.5772/15178.

[2] D. P. Palubiak and M. J. Deen, "CMOS SPADs: Design issues and research challenges for detectors, circuits, and arrays," *IEEE J. Sel. Topics Quantum Electron.*, vol. 20, no. 6, Nov./Dec. 2014, Art. no. 6000718.

[3] A. Tosi and F. Zappa, "MiSPiA: Microelectronic single-photon 3D imaging arrays for low-light high-speed safety and security applications," *Proc. SPIE*, vol. 8899, p. 88990D, Nov. 2013.

[4] D. Bronzi *et al.*, "100 000 frames/s 64×32 single-photon detector array for 2-D imaging and 3-D ranging," *IEEE J. Sel. Topics Quantum Electron.*, vol. 20, no. 6, Nov. 2014, Art. no. 3804310.

[5] I. Vornicu, R. Carmona-Galán, and Á. Rodríguez-Vázquez, "Real-time inter-frame histogram builder for SPAD image sensors," *IEEE Sensors J.*, vol. 18, no. 4, pp. 1576–1584, Feb. 2018.

[6] J. Van de Weijer and R. Van den Boomgaard, "Local mode filtering," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Dec. 2001, pp. II-428–II-433.

[7] F. Villa *et al.*, "CMOS Imager With 1024 SPADs and TDCs for single-photon timing and 3-D time-of-flight," *IEEE J. Sel. Topics Quantum Electron.*, vol. 20, no. 6, Nov./Dec. 2014, Art. no. 3804810.

[8] O. Levy and L. Wolf, "Live repetition counting," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3020–3028.

[9] A. K. Sharma, A. Laflaquiere, G. A. Agranov, G. Rosenblum, and S. Mandai, "SPAD array with gated histogram construction," U.S. Patent 2017 0052 065 A1, Feb. 23, 2017.

[10] S. Lindner, C. Zhang, I. Antolovic, M. Wolf, and E. Charbon, "A 252×144 SPAD pixel flash lidar with 1728 dual-clock 48.8 PS TDCs, integrated histogramming and 14.9-to-1 compression in 180NM CMOS technology," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2018, pp. 9–14.

[11] I. Vornicu, R. Carmona-Galán, Á. Rodíguez-Vázquez, "Live demonstration: Photon counting and direct TOF camera prototype based on CMOS SPADs," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Baltimore, MD, USA, May 2017, p. 1.

[12] N. A. W. Dutton *et al.*, "A time-correlated single-photon-counting sensor with 14GS/s histogramming time-to-digital converter," in *Proc. IEEE ISSCC Dig. Tech. Papers*, vol. 11, Feb. 2015, pp. 1–3.

[13] N. Dutton *et al.*, "Multiple-event direct histogram TDC in 65 nm FPGA technology," in *Proc. Ph.D. Res. Microelectron. Electron.*, Jun./Jul. 2014, pp. 1–5.

[14] Xilinx. *7 Series FPGAs Memory Resources*. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug473_7Series_Memory_Resources.pdf

Ion Vornicu graduated in electronics engineering (specialization in micro-technologies) in 2008 and received the M.Sc. degree in modern signal processing techniques and the Ph.D. degree in microelectronics from the Faculty of Electronics and Telecommunications, Gheorghe Asachi Technical University of Iasi, Romania, in 2011. During the Ph.D. study, his main research focus has been on CMOS implementation of a type of cellular neural networks for spatio-temporal analog image processing, log-domain circuits, smart CMOS imagers. Since 2011, he has been an Associate Researcher with the Institute of Microelectronics of Seville, Spain. His current research interests lie in the design and test of CMOS sensors based on single-photon avalanche diodes used for 3-D vision and nuclear medicine imaging, such as positron emission tomography.

Angela Darie received the B.Sc. degree in electronics engineering (specialization in micro-technologies), the M.Sc. degree in modern signal processing techniques and the Ph.D. degree in electronics and telecommunications from the Faculty of Electronics and Telecommunications, Gheorghe Asachi Technical University of Iasi, Romania, in 2008, 2009, and 2011, respectively. Her Ph.D. thesis focused on implementation of a wireless communication system based on FPGA and WSN motes to estimate a system's propagation characteristics through a propagation medium. Since 2012, she has been an Associate Researcher with the Institute of Microelectronics of Seville, Spain. Her main research areas of interest are digital hardware design and interfaces for data acquisition and control.

Ricardo Carmona-Galán graduated in physics in 1993 and received the Ph.D. degree in microelectronics from the University of Seville, Spain, in 2002. From 1996 to 1998, he was a Research Assistant in Prof. Chua's laboratory with the EECS Department, University of California at Berkeley, Berkeley. From 1999 to 2005, he was an Assistant Professor with the Department of Electronics, University of Seville. Since 2005, he has been a Tenured Scientist with the Institute of Microelectronics of Seville (IMSE-CNM-CSIC). He held a post-doctoral position at the University of Notre Dame, IN, USA, from 2006 to 2007, where he worked in interfaces for CMOS compatible nanostructures for multispectral light sensing. He has collaborated with start-up companies such as Anafocus, Seville, and Eutecus, Berkeley. He has designed several vision chips implementing different focal plane operators for early vision processing. His main research focus has been on VLSI implementation of concurrent sensor/processor arrays for real-time image processing and vision. His current research interests lie in the design of low-power smart image sensors and 3-D integrated circuits for autonomous vision systems. He has co-authored more than 120 papers in refereed journals and conferences and several book chapters. He received the Best Paper Award from the *International Journal of Circuit Theory and Applications*. He was a co-recipient of the award of the ACET and a Certificate of Teaching Excellence from the University of Seville.

Ángel Rodríguez-Vázquez is currently a Full Professor of Electronics with the University of Seville, Seville, Spain, where he is also appointed for research at the Institute of Microelectronics of Seville, Centro Nacional de Microelectrónica, Consejo Superior de Investigaciones Cientficas. He has authored eight books—approximately 50 chapters in edited books, including original tutorials on chaotic integrated circuits, design of data converters, and design of chips for vision—and some 500 articles in peer-reviewed specialized publications. His research work is widely quoted, and he has an h-index of 35. His current research interests are in the areas of imagers and vision systems using 3-D integration technologies and of ultra-low-power medical electronic devices. He has served and is currently serving as an editor, an associate editor, and a guest editor for different IEEE and non-IEEE journals. He is in the committee of many international journals and conferences and has chaired different international IEEE and the Society of Photo-Optical Instrumentation Engineers conferences. He has received a number of international awards for his research work, such as the IEEE Guillemin Cauer Best Paper Award, two best paper awards from the *Wileys International Journal of Circuit Theory and Applications*, the IEEE European Conference on Circuit Theory and Design Best Paper Award, and the IEEE International Symposium on Circuits and Systems Best Demo-Paper Award.