Event-Driven Encoding Algorithms for Synchronous Front-End Sensors in Robotic Platforms

Event-Driven Encoding Algorithms for Synchronous Front-End Sensors in Robotic Platforms

Paolo Motto Ros[2*], *Member, IEEE*, Marino Laterza[1*], Danilo Demarchi[3], *Senior Member, IEEE*, Maurizio Martina[3], *Senior Member, IEEE*, and Chiara Bartolozzi[1], *Member, IEEE*

***Abstract*—Asynchronous, event-driven, sampling techniques adapt the sampling rate of sensory signals to their dynamics, by effectively compressing the data with respect to synchronous, clock-driven, sampling. In robotics such techniques offer data and bandwidth reduction, together with high temporal resolution and low latency. Despite vision and auditory event-driven sensors are currently available, robots are still equipped with a plethora of other sensors that might benefit from the event-driven encoding. In this paper, we study five estimation algorithms that implement event-driven encoding for off-the-shelf clock-driven sensors. Digital accelerometer datasets were used to validate the system in robotic applications; other datasets have been used to assess the general performance of the proposed approach. The two best algorithms in terms of six performance parameters have been implemented on a Xilinx Artix-7 FPGA platform, using 2892 LUTs and 3620 flip-flops and reducing the output bandwidth from -44 % to -75 %, over the considered datasets.**

***Index Terms*—Asynchronous sampling algorithms, Event-Driven, FPGA, Relative Threshold, Output Bandwidth, Robotic Environment.**

I. Introduction

State-of-the-art sensors are mostly based on clock-driven sampling of the physical signal being measured. This approach has a trade-off between the amount of data acquired and the maximum detectable input frequency (Nyquist) of the signal variation. Tuning the clock-rate for very fast signals results in sampling redundant values when the signal is slowly changing, while decreasing the sampling frequency results in missing potentially important signal variations. Additionally, for sensors with multiple sensing sites, such as cameras or large area tactile devices, there is an inherent latency in data transmission, due to the need to synchronously sample all the sensing elements in the device. While the advantage of clock-driven sampling is the compliance of all sensors and acquisition devices to the clock-driven paradigm, the trade-offs and downsides listed above are detrimental for building efficient sensory systems for artificial devices. Specifically, data compression, high temporal resolution and short latency are especially desirable in robotics, where the progressive application in unconstrained scenarios is leading to the integration of an increasing number of sensors needed to perceive both the environment and the status of the robot. A viable alternative is the "neuromorphic" event-driven sensing strategy inspired by biological sensory systems, where the sensing element (e.g. photoreceptors and retinal cells in vision, mechanoreceptors in touch) gets active when it detects a variation in its own input and sends action potentials to neurons in the sensing areas of the brain. The output activity of each sensing neuron encodes for the properties of the sensed stimulus. Similar approaches (eventually sending a sample along with the event) have been investigated and developed in other research areas too, including (but not limited to) automation control and signal processing [4] and energy metering [5].

Neuromorphic event-driven sensing sends data only when the amplitude of the measured signal has experimented a certain change, rather than at fixed time intervals. This change could be referred to the sample which generated the last event [1]. The signal is assumed to stay constant until another event is produced. As a consequence, the reference value could be exploited as a predictor for the future values of the signal. If this estimate differs from the actual sample value by more than a given amount, then a new event is generated. In this encoding scheme, the data is written on the output bus as soon as the change is sensed. In a scenario with many sensing sites (e.g., vision or tactile), this strategy decreases latency dramatically, avoiding the sampling and transmission of the whole set of pixels (or taxels). Information is then encoded in the relative timing between generated events and the value of the sample is not sent, limiting the number of bits to be sent, as opposed to other asynchronous sampling transmissions, which send the whole data sample whenever a threshold-crossing occurs [2]–[4].

This approach resulted so far in the design of event-driven vision [8]–[10], [12], auditory [11] and (more recently) tactile sensing [13], [18], [22], where the sensing element itself implements the data-driven sampling. While event-driven vision sensors have already been integrated on robotic platforms, tactile sensors require further development [21] and other sensor modalities are not yet under development. On the other hand, robots are fully equipped with a plethora of sensors (temperature, pressure, encoders, accelerometers, etc.) and it is possible to emulate event-driven compression using traditional off-the-shelf clock-driven sensors that are readily integrated in robotic environments. The aim of this approach is two-fold: improving efficiency in signal transmission (optimizing

[*] M. Laterza and P. Motto Ros equally contributed to this manuscript

[1] M. Laterza and C. Bartolozzi are with iCub Facility, Istituto Italiano di Tecnologia, Genova, Italy, E-mail: marino.laterza@iit.it, chiara.bartolozzi@iit.it

[2] P. Motto Ros is with Electronic Design Lab, Istituto Italiano di Tecnologia, Genova, Italy, E-mail: paolo.mottoros@iit.it

[3] M. Martina and D. Demarchi are with Dipartimento di Elettronica e Telecomunicazioni, Politecnico di Torino, Torino, Italy, E-mail: maurizio.martina@polito.it, danilo.demarchi@polito.it

bandwidth, data compression, latency, etc.) and delivering prototype systems for the development of event-driven algorithms for perception. An example of emulation of event-driven sensing of clock-based data has been shown in [20], where the event-driven data encoding and transmission have been implemented on an FPGA interfaced to the capacitive sensors of the iCub robot skin [24]. Other applications have been developed in the prosthetic field [14], with the aim of implementing a tactile feedback control, based on the actual data sensed by the prosthesis.

Algorithms proposed for the conversion of clock-sampled data into event-driven show the potential compression performance of this approach. They are based on detection of relative change among current and previous samples. Specifically, the detected change is relative to the absolute value ($^{\Delta x}/_{x}$), de facto implementing a logarithmic compression that increases the compression dynamic range.

In this work, we characterize a set of more complex algorithms, analyzing their performance in terms of data compression and implementation cost. Our goal is to find an algorithm for event-driven encoding that can be applied to any sensory signal acquired by a clock-sampling strategy.

As case study, we tested the algorithms for the encoding of MEMS accelerometers that are integrated in the iCub humanoid robotic platform [24]. As discussed in [20], the mid-term goal is to have a unified tactile/accelerometric sensing system in order to enable the development of event-driven applications — allowing the humanoid robot to interact with the surrounding environment — without requiring the development of new sensors. With this aim, one of the requirement has been to use the same hardware platform (and to respect the same implementation constraints) as done in [20].

The conceived scheme is absolutely general and flexible, so that changing its internal parameters will produce good performance for very different sensors. To offer a thorough analysis of the encoding scheme, we frame it as an estimation problem and compare different solutions.

Starting from the asynchronous algorithm called “Send-on-Delta" [1], where the sample is transmitted when the absolute value of the difference between the current input and the previous one is greater than a given threshold, we also evaluated more complex algorithms. Those algorithms compare the input data with a reference value, in order to decide if an event has to be generated or not. As such, the reference value could be thought of as a predictor of the value of the next sample. For example, in the Send-on-Delta case, the predictor is a zero-order one. As a result, in this alternative view, if the estimation error falls within a given boundary with respect to the measured input, no event (and hence no transmission) is generated. Differently from the standard Send-on-Delta, in the proposed implementation, as soon as the estimation error exceeds the boundary, an event is generated and transmitted. The information is encoded in the exact time at which the event is generated and implicitly transmitted in the timing between events, hence, we do not need to send the absolute value of the sample together with the event. In order to bound the maximum relative error, the change with respect to the predicted sample value is computed using a relative threshold.

The best-performing algorithms are then evaluated in terms of accuracy, output data rate reduction and resource requirement on a Xilinx Artix-7 FPGA (model XC7A35T-L1). Specifically, the accuracy parameter (the error between the original and the reconstructed signal) is used to check if and how much the proposed encoding reduces the information content of the signal, however, in neuromorphic perception the signal is not usually reconstructed and information about the sensed signal is extracted using event-driven algorithms. Also, the compression rate is used to compare the different proposed algorithms, rather than to find the best possible compression strategy for the signal at hand. For this reason, we treated agnostically the signal that we were processing, without using knowledge about the physical origin of the signal itself. That strategy could benefit a specific application, for example by considering the non-independence among the three axis of the accelerometers and the relationship between the position, velocity and acceleration values. In such a case, the compression would be higher, but specific to the accelerometer signal only. Rather, we were looking for a more general algorithm for the event-driven encoding of any sensory signal. In the accelerometer case, the architecture consists in three identical submodules, one per spatial axis, which implement the encoding scheme in parallel. The approach used to send the data on the bus is the so-called Address Event Representation (AER) [15]–[17], where the output data only includes the event polarity (e.g., if the signal is increasing or decreasing with respect to the previous sample) along with the corresponding address of the sensor (in this specific case, of the accelerometer axis). Consequently, the output bus exhibits an asynchronous flow of messages containing the sender address and some bits representing the event polarity information, instead of the whole sample value. As specific AER protocol, we use an implementation of the asynchronous serial AER [15]–[17], that is specifically designed for robotic systems, where minimizing wiring is a strict requirement. We first outline the main differences between the traditional asynchronous sampling schemes and the proposed one (Section II). After setting up the main features of the implemented scheme, the possible event polarities are discussed (Section III). Once the communication system is set, we make a comparison aimed at identifying the best predicting algorithm to embed inside the defined scheme (Section IV). Then, we evaluate the quantitative results of this analysis by means of several performance figures (Section V) and we implement the best-performing algorithms, as well as the communication system, on FPGA (Section VI). The main achievements of the work and future development are finally discussed (Section VII).

II. Event-Generation Scheme

A. Pre-Algorithmic Manipulation of Data

The sampling rate of most commercial digital accelerometers ranges from units to thousands of Hz. It is suitable for a clock-based acquisition from this category of sensors, but an event-driven transmission requires a higher temporal precision. Specifically, the information in event-driven transmission is

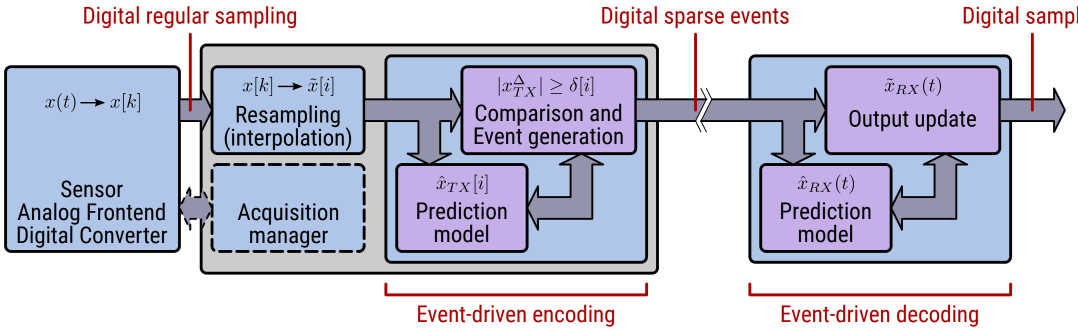


Fig. 1: Overall scheme of the proposed system: the analog signal $x_{TX}(t)$ is first acquired by the sensor and then resampled at a higher $^1/_{T_s}$ rate to improve the timing accuracy; by comparing the state variable x^*_{TX} with the interpolated x'_{TX}, events are eventually generated (EvGen); on the decoding side, a corresponding state variable x^*_{RX} is combined with the reception of events in order to update the output variable x''_{RX}.

triggered exactly by change detection, whereas in clock-based sampling this situation can happen between two consecutive samples.

In the proposed implementation, we internally resample, with a constant period T_s, the data received from the transducer $x[k]$ that corresponds to $x(kT)$ (in the discrete time domain, regularly sampled with a period T) using a linear interpolator and resulting in $\tilde{x}[i]$ that corresponds to $\tilde{x}(iT_s)$ (with the only constraint $\tilde{x}(kT) = x(kT)$, and, usually, $T_s \ll T$). The resampling rate has been fixed at 5 MHz, as in [20], leading to a flow of produced samples every 200 ns. The denser input mimics an analog continuous signal and allows to timely detect the variation of the input signal, therefore increasing the timing precision of the overall acquisition system. The increased timing precision allows for a better reconstruction of the signal, limiting the loss of information that would occur without signal interpolation. The corresponding hardware block will contain a resampling unit, or resampler, feeding the block implementing the event generation (EG).

B. Algorithm Scheme

Figure 1 shows the overall scheme of an event transmitter (TX), including the event generation, and the receiver (RX), including the sample reconstruction. The event generation block computes the estimate for the sample at the next time step — by using one of the algorithms detailed in Section III — and compares it to the sample produced by the resampler. When the absolute value of their difference exceeds a given threshold, an event is generated. Namely, for each point of the linear interpolation, a predicted value is computed: the estimation always depends on the type of the last sent event and it may further depend either on the last value(s) which produced event(s) or on the last estimate(s).

In time-continuous asynchronous sampling algorithms [1]–[4], [6], sampling (and therefore event generation, EvGen) occurs whenever the absolute difference between the predicted value and the input signal (sample) crosses a given (static) threshold, i.e., we can define the sequence $\mathcal{T} \triangleq \{t_i\}$ (with $t_i \in \mathbb{R}^+$ and $i \in \mathbb{N}$) as:

$$\mathcal{T} \triangleq \{t_i \mid |x^{\Delta}_{TX}(t_i)| \geq \delta(t_i) \wedge |x^{\Delta}_{TX}(t_i^-)| < \delta(t_i^-)\} \tag{1}$$

where $\wedge$ refers to the logical AND operation, and we define

$$\begin{aligned} x^{\Delta}_{TX}(t) &\triangleq x_{TX}(t) - \hat{x}_{TX}(t) \\ \delta(t) &\triangleq \delta^* \end{aligned} \tag{2}$$

with $\hat{x}_{TX}(t)$ the predicted value at t, $\delta(t)$ the threshold set to a fixed value δ^*. Since we are dealing with sample-based sensors, with the output data eventually re-sampled or interpolated (at a fixed rate, as described in Sec. II-A), we can define a new sequence $\mathcal{I} \triangleq \{i\}$, corresponding to the sequence $\{t_i = iT_s\}$ (with $i \in \mathbb{N}$) of discrete time-points at which an event is generated, as follows:

$$\mathcal{I} \triangleq \{i \mid |x^{\Delta}_{TX}[i]| \geq \delta[i] \wedge |x^{\Delta}_{TX}[i-1]| < \delta[i-1])\} \tag{3}$$

where we define

$$\begin{aligned} x^{\Delta}_{TX}[i] &\triangleq \tilde{x}_{TX}[i] - \hat{x}_{TX}[i] \\ \delta[i] &\triangleq \delta^* \end{aligned} \tag{4}$$

being $\tilde{x}_{TX}[i]$ and $\hat{x}_{TX}[i]$ the interpolated and predicted, respectively, signals, $\delta[i]$ the threshold set to a fixed (positive) value δ^*.

With this scheme, two events are sufficient to implement an unambiguous communication. We can define a sequence of events $\mathcal{E} \triangleq \{E_i\}$, with each E_i triggered at time $t_i \in \mathcal{T}$ (in case of time-continuous system) or, with the resampling (as in this case), at $i \in \mathcal{I}$, as:

$$E_i = \begin{cases} E_{\uparrow} & \text{if } x^{\Delta}_{TX}[i] \geq \delta[i] \\ E_{\downarrow} & \text{if } x^{\Delta}_{TX}[i] \leq -\delta[i]. \end{cases} \tag{5}$$

$E_{\uparrow}$ and $E_{\downarrow}$ are usually encoded by a polarity bit added to the address of the event [8], [16], [17]. Events are transmitted from the TX to the RX in real-time with a low-latency/low-overhead point-to-point asynchronous AER protocol [19], [20], so that events can be immediately processed by the RX.

On the RX side, whenever an event is received at time t (EvRcv(t)), the corresponding $\tilde{x}_{RX}(t)$ can be updated as follows:

$$\tilde{x}_{RX}(t) = \begin{cases} \hat{x}_{RX}(t^-) + \delta(t) & \text{if EvRcv}(t) = E_{\uparrow} \\ \hat{x}_{RX}(t^-) - \delta(t) & \text{if EvRcv}(t) = E_{\downarrow} \\ \hat{x}_{RX}(t^-) & \text{otherwise} \end{cases} \tag{6}$$

where $\hat{x}_{RX}(t)$ is the predicted value at time t by the RX.

This scheme, even if based simply on a difference, is not optimal in the application under investigation. Indeed, the accelerations provided by the sensor could go as high as 4 G and as low as -4 G, crossing the zero level, as proved in several datasets acquired on the iCub robotic equipment [24]. For an equal relative change (eg. 10% of the reference value), a fixed threshold produces more events for a higher sensor value. However, using fixed thresholds the sensitivity of the event generator is constant for the whole range of the sensor. The advantage of a relative threshold is that the produced number of events is constant for equal relative changes, but the sensitivity decreases for higher sensor values.

In order to obtain an homogeneous event rate over the whole range and to compare several algorithms for equal maximum relative error, a relative-threshold-based scheme has been used, instead. A fixed relative threshold is equivalent to a dynamic absolute one, so in the mathematical model described by equations (3) and (4) we can replace equation (4) with:

$$\delta[i] \triangleq \mu \cdot |\tilde{x}_{TX}[i]| \tag{7}$$

where μ is the relative threshold, in the range $(0, 1)$, and is a constant time-invariant tuning parameter, which is set once. As a consequence, the relative threshold μ describes how the sensitivity of the event generator increases for small sensor values and how the sensitivity decreases for high sensor values. By substituting equation 7 in 3 we can rewrite 3 as:

$$\mathcal{I} \triangleq \{i \mid |x^{\delta}_{TX}[i]| \geq \mu \wedge |x^{\delta}_{TX}[i-1]| < \mu\} \tag{8}$$

where we define

$$x^{\delta}_{TX}[i] \triangleq \frac{\tilde{x}_{TX}[i] - \hat{x}_{TX}[i]}{\tilde{x}_{TX}[i]} \tag{9}$$

so to highlight the relative behavior (w.r.t. $\tilde{x}_{TX}[i]$) of the event generation.

To summarize, an $E_{\uparrow}$ event with a relative threshold μ means that the interpolated value is, in absolute value, 100µ % (or more) higher than the predicted one, whereas an $E_{\downarrow}$ event means that the absolute value of the interpolated sample is 100µ % (or more) lower than the corresponding predicted one. However, thanks to the resampling mechanism, an exact 100µ % difference can be expected. This shall be confirmed by the software simulations on the considered input datasets, which are discussed in Section V.

As a result, the reconstructed signal at the receiver $\tilde{x}_{RX}(t)$ can be obtained as follows:

$$\tilde{x}_{RX}(t) = \begin{cases} \hat{x}_{RX}(t^-) \cdot (1+\mu) & \text{if } \mathrm{EvRcv}(t) = E_{\uparrow} \\ \hat{x}_{RX}(t^-) \cdot (1-\mu) & \text{if } \mathrm{EvRcv}(t) = E_{\downarrow} \\ \hat{x}_{RX}(t^-) & \text{otherwise} \end{cases} \tag{10}$$

where $\hat{x}_{RX}(t)$ is the predicted value at time t by the RX.

Here, when no event is received, the predicted value $\hat{x}_{RX}(iT_s)$ is within $\tilde{x}_{TX}(iT_s) \cdot (1 \pm \mu)$.

As $(1+\mu)$ and $(1-\mu)$ are always positive, $x_{rx}(t)$ will always have the same sign as $x_{rx}(t_{prev})$. Further generalizing, the value of $x_{rx}(t)$ after an arbitrary sequence of (eventually mixed in any order) up and down events is $x_{rx}(t_{after}) = x_{rx}(t_{before}) \cdot (1+\mu)^N (1-\mu)^M$, where N and M are the numbers of up and down events, respectively. As above, given the range of values of μ, there is no value of N and/or M (and therefore no sequence of events) which can lead to have the sign of the reconstructed signal different from that of the initial value, i.e., to have the reconstructed signal ($x_{rx}(t)$) cross the zero. For this reason, we introduce the zero-crossing event.

Algorithm 1: Custom event generation scheme

initialize last event as $E_{\updownarrow}$
$region\_sign = 1$
while *in acquisition* **do**
    Compute $\tilde{x}_{TX}[i]$ by resampling
    if $\tilde{x}_{TX}[i]$ *is outside the base region* **then**
        if *last event was either* $E_{\updownarrow}$ *or* $E_{\downarrow}$ *following* $E_{\updownarrow}$ **then**
            $\hat{x}_{TX}[i] = region\_sign \times \theta$
        else
            $\hat{x}_{TX}[i] \equiv \hat{x}_{\{SoD,Lin,Quad,Avg,PID\}}$
        end
        if *sign of* $\tilde{x}_{TX}[i]$ == *sign of* $\hat{x}_{TX}[i]$ **then**
            if μ *is crossed* **then**
                if *sign of* $x^{\Delta}_{TX}[i] \neq$ *sign of* $\tilde{x}_{TX}[i]$ **then**
                    Transmit $E_{\downarrow}$
                else
                    Transmit $E_{\uparrow}$
                end
            else
                if $\tilde{x}_{TX}[i-1]$ *was inside the base region* **then**
                    Transmit $E_{\downarrow}$
                end
            end
        else
            if $\tilde{x}_{TX}[i-1]$ *was inside the base region* **then**
                $region\_sign = region\_sign \times (-1)$
                Transmit $E_{\updownarrow}$
            end
        end
    else
        if $\tilde{x}_{TX}[i-1]$ *was outside the base region* **then**
            Transmit $E_{\updownarrow}$
        end
    end
end

C. Zero-crossing

The acceleration samples have signed values, as opposed to the capacitance values of a tactile sensor [13] or the grayscale level of a vision sensor [9], [12]. This characteristic, along with the use of a relative threshold, causes a zero-crossing problem to be addressed.

In the communication scheme based on a fixed threshold, the change in the sign of the received value could happen when one out of two possible situations occurs:

1) an $E_{\uparrow}$ event is received when $-\delta(t) < \hat{x}_{RX}(t) < 0$;
2) an $E_{\downarrow}$ event is received when $0 \leq \hat{x}_{RX}(t) < \delta(t)$.

By applying the corrections corresponding to the received events and described in equation (6), the sign of the received signal is changed inherently. As a result, only two types of events are required to obtain an unambiguous scheme including the generation of events with an absolute threshold.

As opposed to that, a relative threshold-based communication scheme does not have this capability, because the correction is performed by means of a multiplication by a positive number. This means that the sign of the estimate cannot change, if only $E_{\uparrow}$ or $E_{\downarrow}$ polarities are used. In addition, if the estimate reaches zero, the following reconstructed values are going to be zero for the rest of the acquisition. This will be referred to as the "sign change problem" in the following.

D. Constraining the relative variation of the input

An additional problem introduced by the relative threshold is that, when approaching zero, the relative variation of the signal is higher and higher. This reduces the effectiveness of the resampling strategy employed immediately after the sensor, because the signal could double or more from one sample to the next (e.g., increasing from 10 mG to 20 mG). Since doubling is equivalent to an increase of 100 %, the expected performance of keeping the relative error under 100μ % cannot be guaranteed (μ is usually set below 1). This will be referred to as the "tracking problem" in the following.

E. Adopted Solution

The tracking problem is solved by the introduction of a base threshold (θ), which prevents the system from producing any event whenever the absolute value of the input signal is under that threshold, as shown in Fig. 2. The value of θ depends on:

- the resolution of the sensor;
- the relative threshold set for the acquisition;
- where and how the sensor is physically connected;
- the external environment.

The base threshold could be further tuned after on-field simulations.

On the other hand, the (relative) data threshold μ should be a trade-off:

1) such to obtain a certain maximum relative error out of the base region;
2) low enough to keep a certain compatibility with respect to the samples produced by the sensor;
3) high enough to avoid that the event-rate could increase significantly.

Once the base region has been introduced, the sign change problem is solved by the use of a third type of event called "Cross Base", $E_{\updownarrow}$, which may be produced in two different situations:

1) the last event produced was an $E_{\uparrow}/E_{\downarrow}$ event;
2) the last event produced was a $E_{\updownarrow}$.

In case 1), the $E_{\updownarrow}$ event communicates to the receiver that the input signal has just gone under the base threshold, causing the estimate to equate $\pm\theta$, where the sign is set equal to the one of the last predicted value. In case 2), a second $E_{\updownarrow}$ after another communicates both the exit from the base region and the change in the sign of the function. Until the next event, the estimate will be $\mp\theta$.

TABLE I: Event coding

Event	Output
$E_{\uparrow}$	10
$E_{\downarrow}$	01
$E_{\updownarrow}$	11
—	00

If, in case 1), after $E_{\updownarrow}$ the signal exits the base region with the same sign, a $E_{\downarrow}$ is produced, to allow the algorithm to restart computing the estimate with the implemented estimation algorithm. This solves the sign change problem without introducing ambiguities in the communication scheme.

Algorithm 1 reports the methodology used to implement the complete communication scheme, including the conditions for the $E_{\updownarrow}$ generation. Because there are three possible events and the no event situation, two output lines are used to code the events, as reported in Table I.

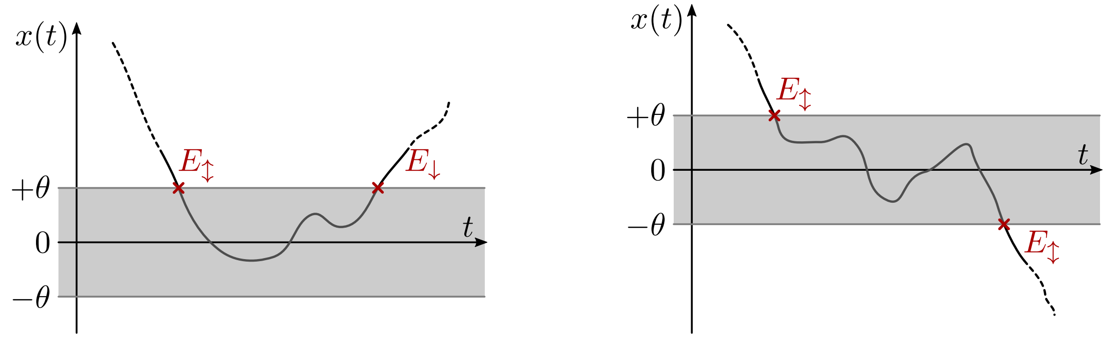
III. Taxonomy of Asynchronous Algorithms

Several asynchronous sampling algorithms have been investigated [1]–[4]. In the application under investigation, the introduction of the dynamic threshold and the choice of sending just events and not the whole sample reduce the quantity of the potentially working methods to the magnitude-driven ones only. At the beginning, even the integral algorithms have been considered. Within this category Send on Area [6] and Send on Energy [7] are well-known methods. However, they are based on an integral relationship with respect to the original signal, so the threshold used concerns the primitive function in the Send on Area and the primitive of the signal squared in the Send on Energy. This prevents the conceived scheme from limiting the maximum relative error on the reconstruction of the signal itself, thus this category of algorithms has not been considered in the following.

On the other hand, the algorithms of the magnitude-driven category are characterized by a direct relationship between the approximating error and the received signal itself. This feature is necessary because all the methods available in the literature are based on a static absolute threshold and on the transmission of the whole data, whereas the main requirement for the system under development is to reduce the output bandwidth. Indeed, with the event-based communication scheme defined in Section II, only 2 bits per spatial axis would be necessary.

Further limitations in the choice of suitable sampling algorithms depend on the need to limit the complexity of the hardware implementation and optimization of resources. After taking into account both the mathematical and complexity criteria, the following algorithms have been analyzed:

A. Send on Delta;
B. Linear;
C. Quadratic;
D. Average;



(a) (b)

Fig. 2: Zero crossing and base threshold θ: when the signal is close to zero, the relative thresholding can cause the generation of way too many events, as the smaller the initial signal is, the finest the sensitivity of the thresholding is. This is solved by setting a “base” threshold. When the signal falls within $+/-\theta$, no events are generated. (a,b) clarify the two cases described in the main text, when the signal changes sign.

E. Proportional-Integral-Derivative (PID).

A. Send on Delta

The Send on Delta (SoD) is the most popular algorithm and uses the last received value corresponding to the last received event, as the estimatefor the following samples, until another event is received.

The approximation $\hat{x}_{SoD}[i]$ yielded by this algorithm corresponds to a zero-order estimation and the formula is independent of the time elapsed from the last event [1]:

$$\hat{x}_{SoD}[i] \triangleq \hat{x}[i] = \hat{x}[i_L] \tag{11}$$

with i_L the index of the last event.

B. Linear

The Linear method is based on a first-order approximation $\hat{x}_{Lin}[i]$ of the signal [2]:

$$\hat{x}_{Lin}[i] \triangleq \hat{x}[i] = \hat{x}[i_L] + \frac{\hat{x}[i_L] - \hat{x}[i_{L-1}]}{i_L - i_{L-1}} \cdot (i - i_L) \tag{12}$$

with i_{L-1} the index of the last but one event.

Differently from the SoD algorithm, the time elapsed from the last and last but one events is used to compute the estimate, so two events are required to compute the estimate. When no event is available or the signal is inside the base region, the estimation is based on the Send-On-Delta algorithm with $\pm\theta$ as an estimate. When only one event is available (e.g., after exiting the base region or after the very first received event), the method uses it as $x^*(t_L)$ and $\pm\theta$ as $x^*(t_{L-1})$.

C. Quadratic

The Quadratic algorithm, characterized by $\hat{x}_{Quad}[i]$, increases the approximation by one order with respect to the Linear method [2]:

$$\begin{aligned}\hat{x}_{Quad}[i] \triangleq \hat{x}[i] = \hat{x}[i_L] + \frac{\hat{x}[i_L] - \hat{x}[i_{L-1}]}{i_L - i_{L-1}} \cdot (i - i_L) + \\ + \frac{1}{2}\left[\frac{\hat{x}[i_L] - \hat{x}[i_{L-1}]}{(i_L - i_{L-1})^2} - \frac{\hat{x}[i_{L-1}] - \hat{x}[i_{L-2}]}{(i_L - i_{L-1}) \cdot (i_{L-1} - i_{L-2})}\right] \cdot \\ \cdot (i - i_L)^2\end{aligned} \tag{13}$$

Like the Linear algorithm, this method has an intrinsic latency, as at least three events out of the base region should have been produced to apply the algorithm.

In addition, not only does this method require to track the time elapsed from the last (i_L) and last but one events (i_{L-1}), it also needs the time elapsed from the last but two event (i_{L-2}). Until two events are available, it behaves like the Linear algorithm; when two events are available, $\hat{x}[i_{L-2}]$ is set to $\pm\theta$.

D. Average

The Average method obtains the estimate$\hat{x}_{Avg}[i]$ with a summation over the last M predicted values, divided by M. When all the last M predicted values are equal to each other, this method is equivalent to the SoD [3]:

$$\hat{x}_{Avg}[i] \triangleq \hat{x}[i] = \frac{1}{M} \sum_{j=0}^{M-1} \hat{x}[i-j] \tag{14}$$

M is chosen depending on the desired number of predicted values one would like to consider and the number of available memory elements, to store the previous predicted values. In the simulations, M has been set to three, as suggested in [3]. If less than M values are available, the average is computed on the available number.

E. PID

The PID algorithm is based on the same theory as in the Control Application field. It groups the Send on Delta, Average and Linear methods (this last one in a further approximate form, to avoid considering the elapsed times), weighing their contributions differently [3] into $\hat{x}_{PID}[i]$:

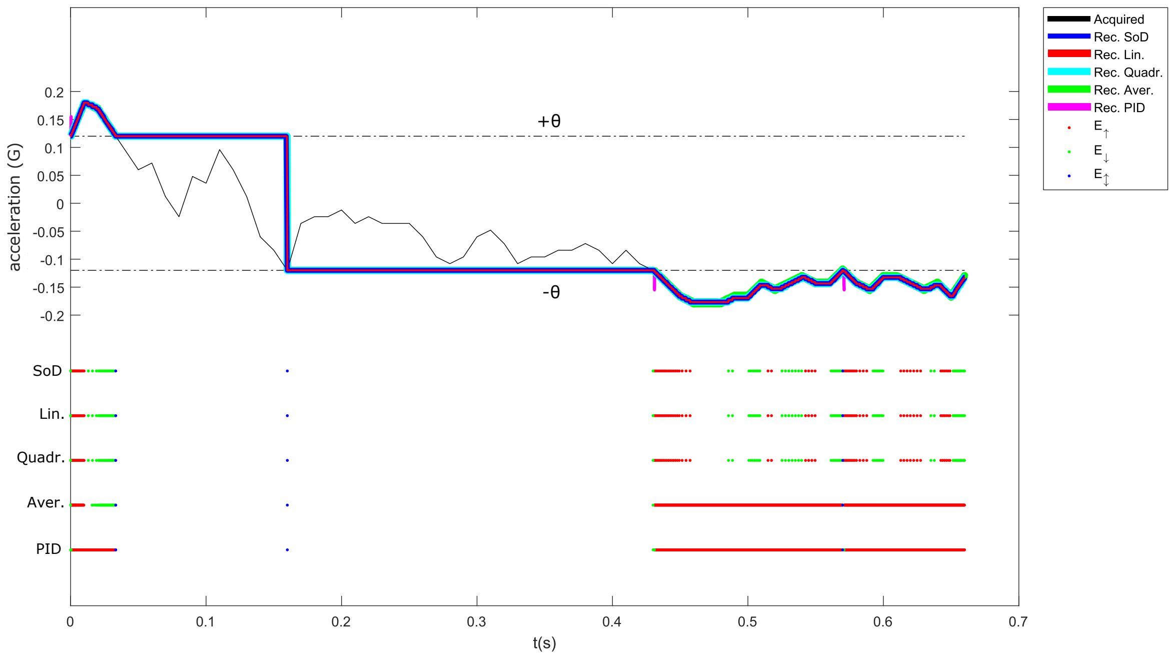


Fig. 3: Event-driven encoding of a signal over time. Top: acquired signal (subset of samples from dataset #8) and reconstructed waveforms using each algorithm, the black lines represent the boundaries of the base region.
Bottom: Events produced by each algorithm with the transmitter fed with the acquired signal. No event is produced during the time interval when the acquired signal is in the base region. At time 0.17 s all the algorithms produce an $E_{\updownarrow}$ when the acquired signal is sampled with opposite sign with respect to the estimate just outside the base region. Consequently, the reconstructed signal changes from $+\theta$ to $-\theta$ (falling edge at 0.18 s).

$$\hat{x}_{PID}[i] \triangleq \hat{x}[i] = w_{SoD} \cdot \hat{x}[i_L] + w_{Aver} \cdot \frac{1}{M} \sum_{j=0}^{M-1} \hat{x}[i-j] + \\ + w_{Lin} \cdot (\hat{x}[i_L] - \hat{x}[i_{L-1}]) \tag{15}$$

As suggested in [3], the three coefficients have been set to:

1) $w_{SoD} = 0.4$;
2) $w_{Aver} = 0.6$;
3) $w_{Lin} = 0.3$.

When less than M events are available, the estimateis computed for each of the three contributions as previously detailed in the descriptions of the corresponding algorithms.

TABLE II: Datasets Used for the Algorithm Test.

Dataset #	Content
1	1^{st} order response
2	2^{nd} order response
3	Electro-cardiography
4-6	Sensor manual static XYZ
7-9	Sensor manual tilting XYZ
10-12	Sensor manual shaking XYZ
13-15	Robot motion XYZ
16-18	Robot static XYZ
19-21	Robot shaking XYZ

IV. Methodology

The algorithms have been written in Matlab® code and they were simulated with twenty-one heterogeneous datasets as input stimuli (Table II). The first three datasets listed were used because already present in the technical literature [2], [3] and to show that the implemented scheme is general and could be employed on control and medical waveforms as well. The datasets related to the sensor were obtained from real-time acquisitions using a general-purpose computer as data collector from the FPGA, which, in turn, receives data from the accelerometer. Three different situations were used for generating the accelerations: a static one, one tilting the sensor along its Z axis and one shaking the sensor along its X axis. The robotic datasets were acquired on the iCub robot left hand in three different movement conditions, similarly to the sensor data: one static, one moving the arm randomly and one shaking the arm.

The events produced have been collected by a software receiver, which reconstructs the original waveform within a maximum relative error equal to the maximum relative threshold set. The reconstructed waveforms, along with the polarity of the received events for a portion of dataset #8, are shown in Figure 3.

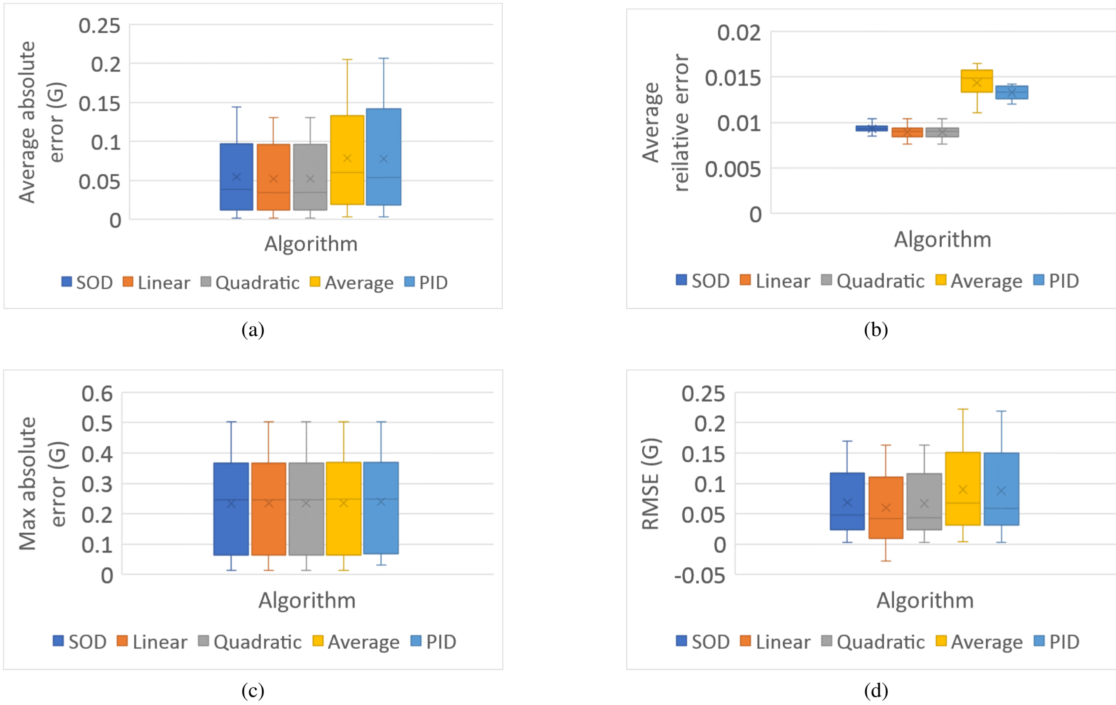


Fig. 4: Results of the simulations: a) γ_{abs}, b) γ_{rel}, c) ξ_{abs} and d) σ.

The performance was evaluated in terms of accuracy, measured as error on the reconstructed waveform (i.e., the error between the $\tilde{x}_{RX}(t)$ computed by the RX and the signal $\tilde{x}_{TX}(t)$ internally used by the TX to generate the events), and effectiveness, measured as the reduction in the number of output messages in the conceived event-driven scheme with respect to a synchronous transmission. For the sake of clarity and without loss of generality, we assume no latency and perfect synchronization (same timings) between TX and RX, so that we can use the two corresponding discrete-time signals $\tilde{x}_{TX}[i]$ and $\tilde{x}_{RX}[i]$. In all the simulations, μ is set to 0.02, in order to have a maximum relative error of 2 %. This value has been chosen as an acceptable tradeoff between reduced data rate and resulting accuracy.

Moreover, following the points listed in Section II-C and the specifications of the employed accelerometer [23], θ is set to 0.12 G.

A. Accuracy figures

The first two performance figures, concerning the accuracy evaluation, have an average meaning: the average absolute error (γ_{abs}) and the average relative error (γ_{rel}) [2].

$$\gamma_{abs} = \frac{1}{N_{samples}} \cdot \sum_{i=n_{in}}^{n_{fin}} |\tilde{x}_{TX}[i] - \tilde{x}_{RX}[i]| \tag{16}$$

$$\gamma_{rel} = \frac{1}{N_{samples}} \cdot \sum_{i=n_{in}}^{n_{fin}} \frac{|\tilde{x}_{TX}[i] - \tilde{x}_{RX}[i]|}{|\tilde{x}_{TX}[i]|} \tag{17}$$

where n_{in} and n_{fin} are the first sample index and the last one, respectively. The average relative error is a factor with respect to the reference. Moreover, the maximum absolute (ξ_{abs}) and relative (ξ_{rel}) errors have been evaluated.

$$\xi_{abs} = max\left(|\tilde{x}_{TX}[i] - \tilde{x}_{RX}[i]|\right) \tag{18}$$

$$\xi_{rel} = max\left(\frac{|\tilde{x}_{TX}[i] - \tilde{x}_{RX}[i]|}{|\tilde{x}_{TX}[i]|}\right) \tag{19}$$

As the signal changes sign, another useful parameter is the root mean squared error (σ):

$$\sigma = \sqrt{\frac{\sum_{i=n_{in}}^{n_{fin}} |\tilde{x}_{TX}[i] - \tilde{x}_{RX}[i]|^2}{N_{samples}}} \tag{20}$$

B. Effectiveness figures

Two performance figures evaluate the gain of the algorithm, where the gain is the number of samples saved by using the event-based method instead of a synchronous one.

The first figure is the equivalent sampling rate (called m in [6]), which is the reciprocal of the average of the inter-event intervals:

$$m = \frac{1}{\Delta t_{aver}} = \frac{N_{events}}{\sum_{i=n_{in}}^{n_{fin}} \Delta t_i} \tag{21}$$

The second figure is called Effectiveness (also called *energy ratio* as in [3]):

$$E = \frac{N_{samples}}{N_{events}} \tag{22}$$

For example, an Effectiveness equal to $\sim 10^3$ means that 1 event every 1000 synchronous samples is sent.

C. Hardware considerations

The maximum available latency is 50 clock cycles because the input frequency equals 5 MHz from the resampler and the internal working frequency of the target system is 250 MHz (both frequencies are compliant with [20]).

V. Simulation Results

As detailed in the previous paragraphs, the proposed communication system is made of different blocks, each of which has been sized as follows. The resampler unit has 16 fractional bits of precision; the integer part depends on the sensor full scale, which had been set to the maximum available on the specific accelerometer employed in this work, i.e, ± 16 G [23]. As a consequence, 6 bits of integer part are required in a signed representation to properly encode even the +16 G value. Moreover, the Linear and Quadratic algorithms also need the inter-event time to be used for the computation, see Eqs. 12) and 13). A 1 s inter-event time could be considered high enough to avoid any timer wrapping, when the acceleration is above the base threshold. In case the acceleration is within the base threshold, a time wrap does not cause any problem because the estimation considers only the base threshold. For a time counter updated at every new sample coming from the resampler, 1 s corresponds to 5 million ticks, i.e., 24 signed bits. For this reason, the integer part of the data is extended to 8 bit and the acceleration data parallelism is set to [8.16] fixed-point representation. As a consequence, the maximum time interval without wrapping is $2^{23} - 1$, about 1.68 s.

Figures 4 and 5 show the performance of the different EG algorithms over all the considered datasets. For the accuracy figures (γ_{abs}, γ_{rel}, ξ_{abs} and σ), the lowest box represents the one which, statistically on the considered datasets, has the best behavior. Conversely, for the effectiveness figures, the situation changes between the equivalent sampling rate (m) and the Effectiveness (E). The lower the equivalent sampling rate, the better, whereas the lower the Effectiveness the worse. As a preliminary evaluation, the maximum relative error has been verified to be under the desired threshold for all the algorithms.

A. Accuracy parameters analysis

In terms of Average Absolute Error, the best algorithms are the Linear and Quadratic ones, with the latter being slightly better than the former. Their values extend from 0.01 G to 0.085 G, approximately. The SoD comes immediately afterwards, whereas the Average and PID have a wider dispersion towards higher values, greater than 0.1 G, with the PID being slightly better than the Average.

In terms of Average Relative Error, the best algorithms are, again, the Linear and Quadratic ones, but in this case the former extends more than the latter in the low-value zone of the error, arriving at 0.4 %, with the upper boundary at 0.7 %. The Send on Delta is again the third best algorithm, exhibiting a very narrow dispersion around 0.9 %. The Average remains the worst, followed by the PID, extending over 1.2 %.

For what concerns the Maximum Absolute Error, the distributions are almost the same for all the algorithms, with the 25^{th} and 75^{th} percentiles between 0.06 G and 0.37 G.

The Root Mean Squared Error shows a situation very similar to the Average Absolute Error case, where the Linear and Quadratic algorithms are very close to each other, with a distribution between 0.02 G and 0.12 G. The Average is still the worst one, followed by the PID and the SoD.

B. Effectiveness parameters analysis

For the effectiveness figures, the Equivalent Sampling Rate shows a very narrow distribution for the Linear and Quadratic algorithms, meaning that their gain is almost independent of the trend of the input signal. The Linear is slightly better because its box extends partially under the one of the Quadratic. The SoD extends more towards high values, until 576 events/s, approximately. The Average remains the worst one even in this case, followed by the PID.

In the end, for the Effectiveness, the first and second order waveforms show that the SoD, Linear and Quadratic have the same performances. However, when considering the remaining 10 datasets a significant difference is shown between the Linear-Quadratic pair and the SoD algorithms: if the former extends between $2.9 \cdot 10^4$ samples/event and $7.0 \cdot 10^4$ samples/event, the latter box is comprised between $2.1 \cdot 10^4$ samples/event and $4.8 \cdot 10^4$ samples/event. The SoD is below, and the Average and PID methods are even lower than the other ones.

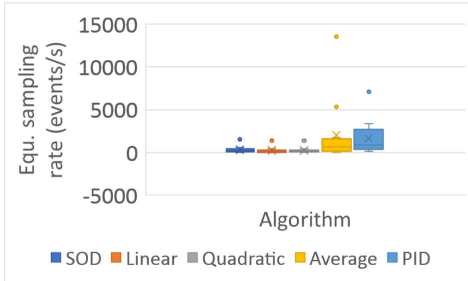
More in detail, the performance of the SoD equals the ones of the Linear and Quadratic methods for the datasets acquired with a manual movement of the accelerometer. On the other hand, the Linear and Quadratic algorithms are always better than the SoD on the robotic datasets, with:

- Accuracy parameters: -0.4 % up to -26.7 % over SoD (-5.4 % avg.);
- Equivalent sampling rate: -7 % up to -58 % over SoD (-20.0 % avg.);
- Effectiveness: +8 % up to +142 % over SoD (+48.0 % avg.).

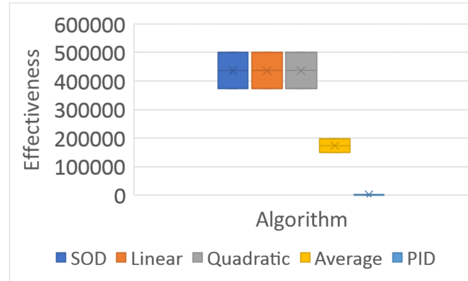
This analysis allows to conclude that the best methods are the Linear and Quadratic, followed by the SoD. As the other two algorithms are more complex and show worse performance than the SoD, they are not addressed in the following part of this work.

VI. Hardware Implementation

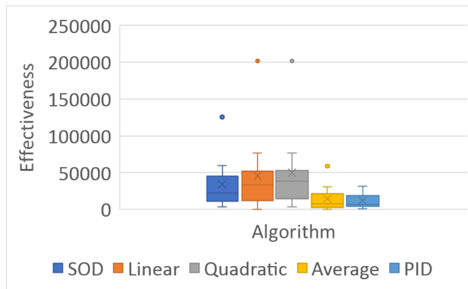
The conceived algorithmic scheme has been described in VHDL to be implemented onto an FPGA platform. In order to achieve the target operating frequency, pipeline stages have been added inside the hardware block. The number of pipeline stages from input to output is however limited from the 50 clock cycles latency requirement. The available FPGA is a speed-grade 1, the lowest speed-grade for the Xilinx Artix-7 XC7A35T model. As a consequence, the number of pipeline stages was tuned until the target frequency was met and the latency requirement was not exceeded (see Section IV-C). This led to a $\frac{N}{2}$ pipe stages inside the multipliers and N inside the divider employed in the Linear and Quadratic algorithms, with $N = 24$ as detailed in Section V. While for the Linear the total latency in the worst case (the one requiring the estimation correction) is within 50 clock cycles, the Quadratic exceeds



(a)



(b)



(c)

Fig. 5: Results of the Matlab® simulations: a) Equivalent Sampling Rate (m), b) Effectiveness related to the first two datasets and c) Effectiveness for all the remainder datasets.

this limit. Moreover, the performance parameters show that the 2nd order term does not give any appreciable advantage over the Linear algorithm. These two considerations allow to discard the Quadratic algorithm.

Finally, considering both the algorithmic performance and hardware optimization, the Linear and Send-on-Delta algorithms appear the most suitable to be mapped onto the FPGA.

Since the employed transducer is a tri-axis accelerometer, the hardware blocks implementing the resampling and the estimation have been replicated once per each axis. The complete architecture is shown as a block diagram representation in Figure 6. The SPI Manager block acquires the synchronous samples from the accelerometer by using a 4-wire SPI communication. Then, it splits the data of each axis and feeds them to the corresponding resampling block. The output event lines consist, each, in a 6-bit address that identifies the axis and the accelerometer, followed by the 2-bit event information in the LSBs. Keeping the address inside the transmitted data is useful in case the event stream from the accelerometer shares the output channel of the system, where it is embedded, with other event streams coming from different sensors. This allows the receiving end to acknowledge the source of the arriving events and is based on the Address-Event-Representation (AER) [15]–[17]. Moreover, if the 3 event lines are multiplexed at the transmitter, only 8 bits instead of 24 are present, further reducing the routing cost.

Figure 7 shows the hardware arrangement. The MEMS accelerometer is connected to an Arty board, hosting an Artix-7 FPGA. The connection is achieved by using a series of jumpers toward one of the PMOD connectors of the Arty board. The board is also connected to a general-purpose computer with two cables: the black one in Figure 7 is a USB cable, used to program the FPGA and to receive the raw acceleration samples from the FPGA; the white one is an Ethernet cable, for collecting the events from the FPGA. The Ethernet payload contains the data coded in AER.

A. Results

Table III shows the post-implementation complexity (estimated by Xilinx Vivado® software) for the single EG block when it implements either the SoD or the Linear algorithm. By replicating it 3 times and adding the SPI Manager block and the resamplers, the LUT usage is 2892 elements (13.9 % of the total) and the FF usage is 3620 elements (8.7 % of the total) for the Linear case. The block works reliably at the target frequency of 250 MHz.

In order to evaluate the improvement with respect to a synchronous system, the output bandwidth is estimated as:

$$BW_o = o_b \sum(m) \tag{23}$$

where o_b is the number of output bits and $\sum(m)$ is the sum of the equivalent sampling rates obtained on each axis, due to the multiplexed output. It is substituted with the fixed output data rate of the accelerometer in the synchronous system.

B. Hardware setup

The synchronous system has a multiplexed output, for a symmetrical comparison to the event-driven one. As a result, the synchronous system has an 18-bit output data (6-bit address and 12-bit acceleration data). The worst-case situation for the Manual acquisitions is the shaking case (datasets 10-12) with $\sum(m) = 753.9\,\text{events}/\text{s}$. The worst-case situation for

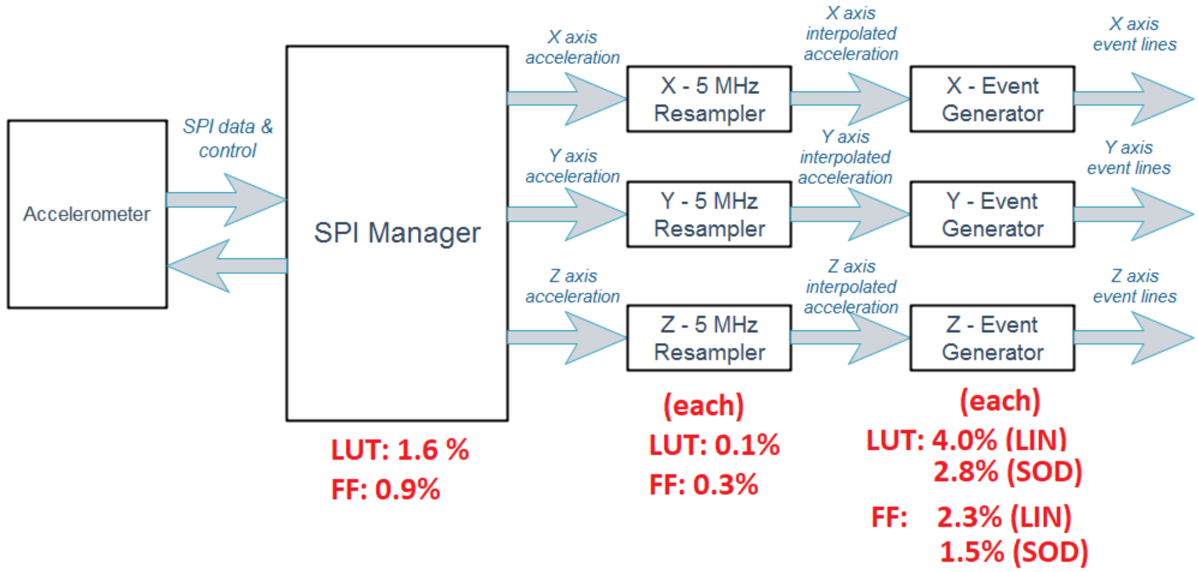


Fig. 6: Block diagram of the complete hardware architecture and relative resource usage of every block after implementation on a Xilinx XC7A35T-L1 FPGA.

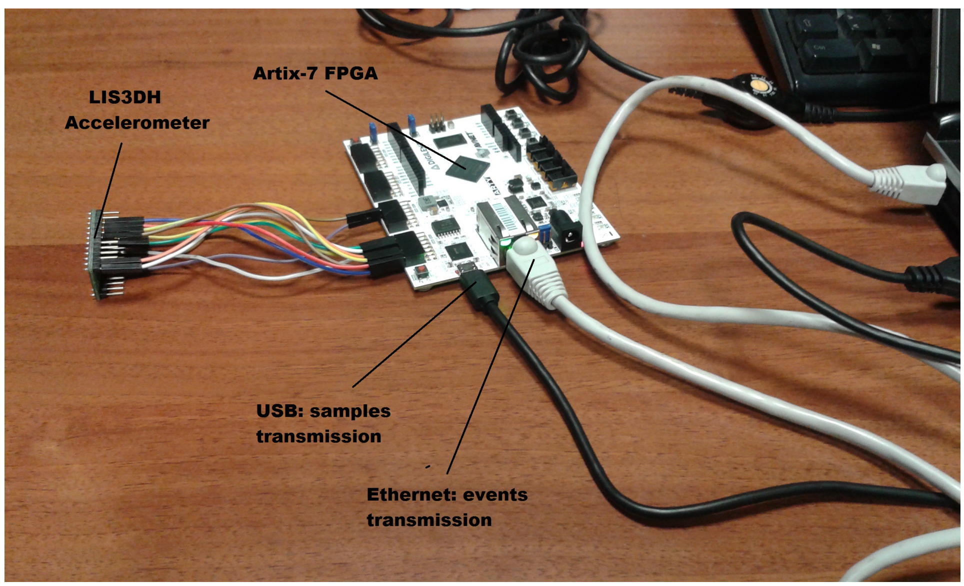


Fig. 7: Hardware setup for the real-time acquisition.

the Robotic datasets is the static case (datasets 13-15) with $\sum(m) = 1698.0$ events/s. The synchronous data, corresponding to the previous two categories of datasets, are produced by the accelerometer at 1344 Hz.

Figure 8 shows the reconstructed waveforms using SoD and Linear algorithms on ten out of the twenty-one considered datasets.

The performance of the algorithmic block on a single-output data are also considered for the first 3 datasets, which are generic waveforms. In that case the worst-case equivalent sampling rate is 229.5 events/s, obtained with the Medical waveform (dataset #3, Figure 8c). The synchronous data rate for that dataset is 360 Hz. The BW_o variation is shown in Table IV, when switching from a synchronous transmission to an event-driven one operating on the same input datasets.

As it can be observed, when the movement has a full

TABLE III: Complexity of the implemented algorithms on Xilinx XC7A35T-L1 FPGA.

Block/Port	SoD	Linear
LUTs	582 (2.8 %)	842 (4.0 %)
FF	624 (1.5 %)	969 (2.3 %)
BRAM	0 (0 %)	0 (0 %)
IO	30 (14.3 %)	30 (14.3 %)
BUFG	2 (6.3 %)	2 (6.3 %)
MMCM	1 (20 %)	1 (20 %)

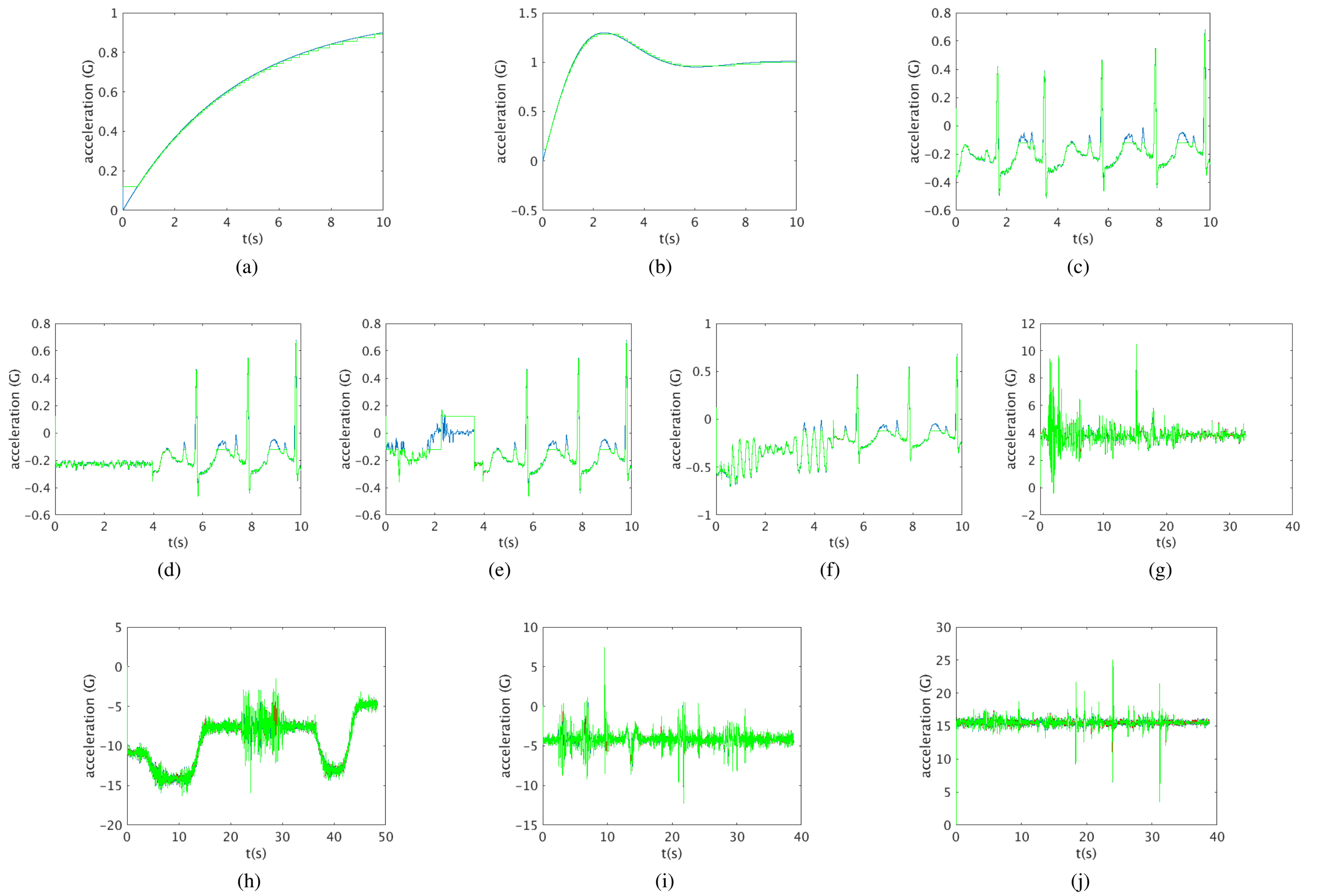


Fig. 8: Example of the considered datasets. the waveforms include the input resampled data (blue), the reconstructed signal with a Linear algorithm (green), the reconstructed signal with a SoD algorithm (red) : a) dataset #1, b) dataset #2, c) dataset #3, d) dataset #4, e) dataset #7, f) dataset #10, g) dataset #14, h) dataset #16, i) dataset #19, j) dataset #21.

TABLE IV: Output Bandwidth Comparison.

Datasets	Synchr. (kbps)	Event-dr.(kbps)	Difference
Manual	24.2	6.0	-75 %
Robotic	24.2	13.6	-44 %
CNTRL & Medical	6.5	1.8	-72 %

dynamic within $\pm 2\,$G, i.e., for a manual movement, the SoD yields the same performance as the Linear, being very attractive for its reduced complexity. In the robotic equipment, instead, the Linear algorithm performs better as discussed in Section V.

VII. Conclusion

This paper has detailed the entire process aimed at conceiving a new event-driven scheme, identifying the best event-generation algorithm from accuracy, effectiveness and implementation points of view. Both the communication scheme and the algorithm were mapped onto an FPGA hardware to code the information received from a digital MEMS accelerometer. Experimental results show that the best estimation algorithm is the first-order, or Linear, one. The real-time acquisitions show that the maximum relative error is kept bounded within the desired relative threshold set at the beginning of the acquisition. The total resource usage in a low-cost FPGA does not exceed the 15 % of both the logic cells and flip-flops. The output bandwidth, thanks to both the 8-bit output event coding and the obtained Effectiveness, is reduced by more than 40 % in all the acceleration datasets, with a -44 % improvement in the robotic platform case. An improvement of -72 % is also observed for single waveforms with slower synchronous data rate.

Although the analysis of the event-driven generation methods has been performed using an accelerometer as input device, the formulation, characterization and FPGA implementation are general enough to hold for different types of sensors, as needed in a fully event-driven robotic sensing system. The main advantages of the event-driven approach are low latency and compression. Our accuracy results show that the compression does not decrease the information content gathered by the sensor. The challenge in the design of artificial perception based on this principle is that of finding principled methods to extract this information without resorting to signal reconstruction and "traditional" algorithms. A fair amount of work has been published so far on the development of such methods, mostly on vision for robots. All the methods and results reported so far show that not only "frame" reconstruction is unnecessary, but also that those algorithms are robust to noise and to the loss of few events (hence having some drop in accuracy). This work goes in the direction of making other sensory modalities available to roboticists to develop multi-modal perception systems that improve efficiency, robustness and autonomy of robots, developing event-driven multi-sensory perception algorithms, ready for when native event-driven sensors will be mature enough to be integrated in robots. In the specific case of accelerometers, the information gathered from the sensor will be useful to assess the movement of the robot, or to detect impact on surfaces, or to classify roughness of surfaces from vibrations. In the first case, the latency of the signal is crucial to detect the contact and correct the action; in the second case, the frequency content of the signal, rather than the acceleration instantaneous value, is important. To study all these aspects, future work includes the integration of the designed system into the iCub robot and the substitution of the Ethernet output with the custom serial protocol discussed in [20], for event transmission in AER packets.

Additionally, the design is fully-portable to other platforms, like ASICs. We will hence develop custom chips with event-driven event generation for off-the-shelf sensors to reduce size and power consumption due to the use of the FPGA. Further developments include the implementation of neural algorithms instead of asynchronous sampling ones and the design of an event-driven readout accelerometer which uses the conceived communication scheme and algorithm internally.

Authors' Notice

The material (i.e., the source code and datasets) presented in this work could be provided by the authors upon request.

References

[1] M. Miskowicz, "Send-on-delta concept: An event-based data reporting strategy", in *Sensors*, v. 6, pp. 49-63, 2006.

[2] K. Staszek, S. Koryciak, M. Miskowicz, "Performance of send-on-delta sampling schemes with prediction", in *Proceedings of IEEE International Symposium on Industrial Electronics ISIE 2011*, pp. 2037-2042, June 2011.

[3] F. Xia, Z. Xu, L. Yao, W. Sun, M. Li, "Prediction-based data transmission for energy conservation in wireless body sensors", in *Proceedings of Wireless Internet Conf. WICON 010*, pp. 1-9, March 2010.

[4] Various authors, *Event-Based Control and Signal Processing*, ed. By M. Miskowicz, CRC Press, 2016.

[5] M. Simonov, G. Chicco and G. Zanetto, "Event-Driven Energy Metering: Principles and Applications", in *EEE Transactions on Industry Applications*, vol. 53, no. 4, pp. 248–251, July-Aug. 2017. doi: 10.1109/TIA.2017.2679680

[6] M. Miskowicz, "The event-triggered integral criterion for sensor sampling", in *Proceedings of IEEE International Symposium on Industrial Electronics ISIE 2005*, v. 3, pp. 1061-1066, November 2005.

[7] M. Miskowicz, "Efficiency of event-based sampling according to error energy criterion", in *Sensors*, v. 10, n. 3, pp. 2242-2261, 18 Mar. 2010.

[8] Posch, C., Matolin, D. and Wohlgenannt, R., "A QVGA 143dB Dynamic Range Asynchronous Address-Event PWM Dynamic Image Sensor with Lossless Pixel-Level Video Compression", in *2010 IEEE International Conference on Solid-State Circuits (ISSCC)*, 7-11 Feb. 2010.

[9] Posch, C., Matolin, D. and Wohlgenannt, R., "An asynchronous time-based image sensor", in *2008 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 2130-2133, 18-21 May 2008.

[10] P. Lichtsteiner, C. Posch and T. Delbruck, "A 128×128 120dB 30mW asynchronous vision sensor that responds to relative intensity change", in *2006 IEEE International Solid State Circuits Conference - Digest of Technical Papers*, San Francisco, CA, 2006, pp. 2060-2069.

[11] V. Chan, S.-C. Liu and A. van Schaik, "AER EAR: A matched silicon cochlea pair with address event representation interface", in *IEEE Transactions on Circuits and Systems I*, 2007, pp. 48-59, vol. 54(1).

[12] B. Son, Y. Suh, S. Kim, H. Jung, J. Kim, C. Shin, K. Park, K. Lee, J. Park, J. Woo, Y. Roh, H. Lee, Y. Wang, I. Ovsiannikov and H. Ryu, "4.1 A 640×480 dynamic vision sensor with a 9um pixel and 300Meps address-event representation", in *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, CA, 2017, pp. 66-67.

[13] E. Baglini, G. Cannata and F. Mastrogiovanni, "Design of an embedded networking infrastructure for whole-body tactile sensing in humanoid robots"in *10th IEEE-RAS International Conference on Humanoid Robots*, 2010.

[14] L. Osborn, R. R. Kaliki, A. B. Soares and N. V. Thakor, "Neuromimetic event-based detection for closed-loop tactile feedback control of upper limb prostheses", in *IEEE Transactions on Haptics*, vol. 9, no. 2, pp. 196–206, April 2016.

[15] M. Sivilotti "Wiring Considerations in analog VLSI Systems with Application to Field-Programmable Networks", 1991.

[16] A. Linares-Barranco, G. Jimenez-Moreno, A. Civit-Ballcels and B. Linares-Barranco, "On synthetic AER generation", in *2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No.04CH37512)*, 2004, pp. V-784-V-787 Vol.5.

[17] R. Paz-Vicente, A. Linares-Barranco, A. Jimenez-Fernandez, G. Jimenez-Moreno and A. Civit-Balcells, "Synthetic retina for AER systems development", in *2009 IEEE/ACS International Conference on Computer Systems and Applications*, Rabat, 2009, pp. 907-912.

[18] P. Motto Ros, M. Crepaldi, and D. Demarchi, "A hybrid quasi-digital/neuromorphic architecture for tactile sensing in humanoid robots", in *Advances in Sensors and Interfaces (IWASI), 2015 6th IEEE International Workshop on Advances in Sensors and Interfaces*, June 2015.

[19] P. Motto Ros, M. Crepaldi, C. Bartolozzi and D. Demarchi, "Asynchronous DC-free serial protocol for event-based AER systems", in *2015 IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, Cairo, 2015, pp. 248–251. doi: 10.1109/ICECS.2015.7440295

[20] C. Bartolozzi, P. Motto Ros, F. Diotalevi, N. Jamali, L. Natale, M. Crepaldi and D. Demarchi, "Event-driven encoding of off-the-shelf tactile sensors for compression and latency optimisation for robotic skin", in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, 2017, pp. 166-173.

[21] S. Caviglia, L. Pinna, M. Valle, and C. Bartolozzi, "An event-driven POSFET taxel for sustained and transient sensing", in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 349-352, May 2016.

[22] F. Bergner, E. Dean-Leon, and G. Cheng, "Event-based signaling for large-scale artificial robotic skin — realization and performance evaluation", in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*, October 2016.

[23] ST Microelectronics, *LIS3DH - MEMS digital output motion sensor: ultra-low-power high-performance 3-axis "nano"accelerometer*, December 2016.

[24] A. Parmiggiani, M. Maggiali, L. Natale, F. Nori, A. Schmitz, N. Tsagarakis, J. Victor, F. Becchi, G. Sandini, G. Metta, "The design of the iCub humanoid robot", in *International Journal of Humanoid Robotics*, vol. 09, no. 04, 1250027, 2012.

Marino Laterza is Fellow researcher at Istituto Italiano di Tecnologia. He received the Bachelor's Degree in Electronic Engineering (Summa cum Laude) in 2015 and the Master of Science in Electronic Engineering in 2017 (Summa cum Laude), both from Politecnico di Torino, Italy. His interests regard digital hardware design on both programmable and application-specific platforms. His current research is focused on the implementation of neural models on FPGA to emulate human perception in tactile sensing.

Paolo Motto Ros Paolo Motto Ros is Senior Post-Doc researcher at Istituto Italiano di Tecnologia, Electronic Design Laboratory. He received the electronic engineering degree and the Ph.D. in electronic engineering from the Politecnico di Torino (Polito), Turin, Italy, in 2005 and 2009, respectively. From 2009 to 2012 he was with Neuronica Laboratory (Dipartimento di Elettronica, Politecnico di Torino) as Post-Doc, working on assistive technologies, computer vision and learning machines projects. He joined the Center for Space Human Robotics (CSHR), Istituto Italiano di Tecnologia (IIT), Turin, Italy, in 2012; since 2016 he is with Electronic Design Laboratory (EDL), IIT, Genoa, Italy. He is an IEEE member since 2016, and member of the Circuits And Systems (CAS) society. He counts >20 publications; current research include design of full-custom ultra-low-power asynchronous digital integrated circuits, event-based smart-sensors, sensor networks, bio-inspired electronics, neuromorphic engineering.

Danilo Demarchi Associate Professor at Politecnico di Torino, Italy, Department of Electronics and Telecommunications. Lecturer at EPFL Lausanne, Switzerland, Adjunct Professor at Tongji University Shanghai, China, Associate Faculty at the University of Illinois at Chicago, Department of Electrical and Computer Engineering. Leading the MiNES (Micro&Nano Electronic Systems) Laboratory of Politecnico di Torino. Senior Member of IEEE, Member of the BioCAS Technical Committee, Associate Editor of the Transactions on Biomedical Circuits and Systems (TBioCAS), Associate Editor of IEEE Sensors and of the Springer Journal BioNanoScience. General Chair of BioCAS 2017 (Biomedical Circuits and Systems) Conference edition in Torino.

Maurizio Martina Maurizio Martina (IEEE Senior Member) is associate professor at Politecnico di Torino. He received the electronic engineering degree and the Ph.D. in electronic engineering from the Politecnico di Torino in 2000 and 2004, respectively. From 2004 he is with the VLSI-lab (Electronics and Telecommunications Department, Politecnico di Torino) working on VLSI architectures for digitial signal processing. He is author and coauthor of about 100 scientific papers.

Chiara Bartolozzi Chiara Bartolozzi (IEEE Member) is researcher at the Istituto Italiano di Tecnologia. She earned a degree in Engineering (with honors) at University of Genova (Italy) and a Ph.D. in Neuroinformatics at ETH Zurich, developing analog subthreshold circuits for emulating biophysical neuronal properties onto silicon and modelling selective attention on hierarchical multi-chip systems. She is currently leading the Event Driven Perception for Robotics group (www.edpr.iit.it), mainly working on the application of the "neuromorphic" engineering approach to the design of sensors and algorithms for robotic perception. She is chair of the Neuromorphic Systems and Application Technical Committee of IEEE CAS.