


Article

# Movement Detection with Event-Based Cameras: Comparison with Frame-Based Cameras in Robot Object Tracking Using Powerlink Communication

Juan Barrios-Avilés <sup>†</sup>, Taras Iakymchuk <sup>†</sup>, Jorge Samaniego <sup>†</sup> and Leandro D. Medus <sup>†</sup>  
and Alfredo Rosado-Muñoz <sup>\*,†</sup> 

GDDP. Department Electronic Engineering, School of Engineering, Universitat de Valencia, Burjassot, 46100 Valencia, Spain; juan.barrios@uv.es (J.B.-A.); taras.yakymchuk@uv.es (T.I.); jorsale@gmail.com (J.S.); leandro.medus@ext.uv.es (L.D.M.)

\* Correspondence: alfredo.rosado@uv.es; Tel.: +34-963-543-808

† These authors contributed equally to this work.

Received: 3 October 2018 ; Accepted: 2 November 2018; Published: 7 November 2018



**Abstract:** Event-based cameras are not common in industrial applications despite the fact that they can add multiple advantages for applications with moving objects. In comparison with frame-based cameras, the amount of generated data is very low while keeping the main information in the scene. For an industrial environment with interconnected systems, data reduction becomes very important to avoid network congestion and provide faster response time. However, the use of new sensors as event-based cameras is not common since they do not usually provide connectivity to industrial buses. This work develops a network node based on a Field Programmable Gate Array (FPGA), including data acquisition and tracking position for an event-based camera. It also includes spurious reduction and filtering algorithms while keeping the main features at the scene. The FPGA node also includes the stack of the network protocol to provide standard communication among other nodes. The powerlink IEEE 61158 industrial network is used to communicate the FPGA with a controller connected to a self-developed two-axis servo-controlled robot. The inverse kinematics model for the robot is included in the controller. To complete the system and provide a comparison, a traditional frame-based camera is also connected to the controller. Response time and robustness to lighting conditions are tested. Results show that, using the event-based camera, the robot can follow the object using fast image recognition achieving up to 85% percent data reduction providing an average of 99 ms faster position detection and less dispersion in position detection (4.96 mm vs. 17.74 mm in the Y-axis position, and 2.18 mm vs. 8.26 mm in the X-axis position) than the frame-based camera, showing that event-based cameras are more stable under light changes. Additionally, event-based cameras offer intrinsic advantages due to the low computational complexity required: small size, low power, reduced data and low cost. Thus, it is demonstrated how the development of new equipment and algorithms can be efficiently integrated into an industrial system, merging commercial industrial equipment with new devices.

**Keywords:** event-based camera; event-based processing; Powerlink bus; Powerlink FPGA controlled node; object tracking; two-axis robot

## 1. Introduction

The amount of data transmitted through communication networks is increasing at a higher pace than the supported bandwidth. Especially in industrial environments where real-time and low-latency systems are required, the saturation of communication networks due to the addition of advanced equipment generating and transmitting a high amount of data can be a problem [1]. Event-based

cameras produce data in the form of asynchronous events [2], and data are only generated when there is a difference in light intensity (defined by a threshold) received by any of the sensors in the camera (pixels) arranged in an array. The generated event includes information about the address of the pixel in the sensor where the threshold was exceeded, together with a time-stamp in order to generate a unique event, not just in space but also in time. It is possible to define if the event is caused by an intensity increment or a decrement, causing a positive or negative event. This behaviour is similar to a mammal brain [3], which leads to using neuromorphic systems [4,5] for further information processing, feature extraction, scene detection [6] and filtering [7,8]. Proper lighting is a key factor in traditional industrial vision systems since it is difficult to maintain a constant light due to a constantly changing environment. Traditional solutions required the use of specific lighting systems suited for specific applications [9–11]. Event-based cameras minimize light effects since only pixel intensity differences are considered and no need of specific light intensity is required, independently of light conditions.

Currently, applications working with event-based cameras have been developed for research purposes, emulating a neuromorphic system, and only a few are targeting the industrial sector [12]. Some works are focused on developing and improving systems for data exchange between two or more bioinspired devices [13,14], with applications in fields as medicine or biology. Event-based data attracted the interest since basic information could be reported by using a reduced data size when compared, for instance, with periodical data sampling. This fact allows the simplification of computation, reduces power and data size. Events can be created when a delta difference in the read value is achieved, first introduced by Miskowicz [15] and later used in different forms for tracking [16], mobile robots control [17,18], or optimization for the feedback control in smart sensors [19].

However, event-based systems have not yet achieved the desirable spread in industrial environments to benefit from their advantages. Event-based cameras were conceived according to the models of early visual processing in biological systems [20,21]. They now form part of the neuromorphic systems [22], devoted to all electrical and computer-based sensing, processing and actuation taking biology behaviour as its form of operation.

On the other hand, neuromorphic processing techniques are focused on producing more and better data for pattern recognition in neuromorphic systems [23,24] and develop machine learning algorithms for classification, prediction or recognition [25,26]. However, previous processing of event-data before entering the machine learning can ease the task by removing artifacts or irrelevant scene information. Taking this into consideration, the authors developed an algorithm called LDSI (Less Data Same Information) which was designed and tested for processing and filtering data from event-based cameras, achieving high data reduction ratios and keeping the main relevant information at the scene [27]. The algorithm processes on-off event sequences mimicking biological neurons, being fully configurable to provide adjustable results of filtering and data reduction depending on the final application: event rate, noise, image size and light conditions, amongst others. The LDSI algorithm has a low computational complexity, requiring low power for computation in the same network node as the event generator unit, reducing data transfer in the network, lower than frame-based cameras, and improving the response time of the overall system. The proposed approach is aimed at globally lower the computational burden and use reduced device resources, which is a very important issue for decentralized industrial systems.

Nowadays, it is clear that Ethernet has taken the lead in industrial communications, offering excellent price, performance and robustness capabilities. Ethernet has been consolidated as a solid framework for information exchange in all industrial levels, from management to the industrial plant. Moreover, with the advent of Industrial Internet of Things (IIoT) linked with the Industry 4.0 concept, Ethernet communication plays a key role in the deployment of efficient machines, data collection and control strategies in the next generation factories. Different protocols use Ethernet as a common physical connection (mainly, layers 1 and 2 according to the ISO/OSI model). The widely spread TCP/IP is very well known and can be used for data management and supervision in industrial levels, but it is not deterministic and cannot be used in the industrial plant where controllers, sensors and

actuators must exchange information in a very short time, and even more important, in a maximum guaranteed time so that fast industrial processes can be properly run, i.e., in a deterministic time-frame. In this scenario, multiple Ethernet-based protocol proposals exist, claiming to be the fastest, more robust and industry qualified. Some examples of the most popular protocols are Ethernet/IP [28], Profinet [29], EtherCAT [30] and powerlink [31]. All of these protocols are based on Ethernet with some modifications introduced in different layers from the ISO/OSI levels so that additional data handling techniques are added to satisfy timing requirements, data traffic in the bus, etc., when compared to the widely used TCP/IP protocol.

Current neuromorphic event-based systems still use a high bandwidth to transmit data, higher than typical industrial systems could handle, making their advantages being overshadowed and making conventional frame-based machine vision systems still being the used vision technology in industry. According to [32], an advanced event-based sensor with about 1 million neurons might generate up to  $10^8$  million events per second; in a relatively simple example, they show an experimental test where 8 million events per second are generated. When transmitting such amount of information into a network shared with other nodes, the risk of saturation is high. Additionally, determinism is essential in industrial networks and thus, especially in the IIoT era where many network nodes exchange data at high speed, bus saturation must be avoided [33]. Several authors studied the effect of data saturation in Computer Numerical Control (CNC) machines [34] and how reliability and real-time performance may be degraded under intensive use of the network [35].

The aim of this work is to develop a network node for event-based cameras including data processing for high speed tracking, showing the feasibility of developing new nodes for Ethernet networks to deploy a full industrial system including commercial equipment and also non-commercial additions: new hardware as an event-based camera and a self-developed two-axis positioning robot and related processing algorithms as movement detection and network protocol implementation.

To investigate the feasibility of integrating event-based systems into modern industrial systems, a resource- and latency-demanding task of tracking a fast-moving object is proposed. We developed a fast, accurate and low data transfer algorithm for object tracking using an event-based camera. The tracking of fast-moving objects is not an easy task for frame-based cameras since the movement can be faster than the frame rate, which results in missing object positions. A typical industrial solution is the use of high speed cameras with a high frame rate, increasing data flow and requiring significant computational resources. The use of an event-based camera is a viable alternative since it can provide accurate tracking at any speed with low data flow and computational burden.

A network node based on an FPGA was developed. It includes event-based camera data retrieval, event processing algorithms for filtering noise from the camera, object tracking position algorithm and powerlink data transfer protocol to serve as a controlled node. In order to prove powerlink capabilities, the FPGA is integrated into a network including a managing node and two more controlled nodes: a two-axis servo controller and a distributed I/O unit based on a Programmable Logic Controller (PLC). The controller manages the communication among powerlink nodes but also includes the real-time computation of the inverse kinematics for a self-developed two-axis robot made with two synchronous motors. Additionally, a MODBUS/TCP communication with an industrial PC connected to a frame-based industrial vision system for object tracking with traditional computer vision techniques is used for experimental comparison with the event-based camera. In addition to less computational requirements and data reduction, experimental tests show that the event-based camera is able to provide faster response time and is more robust to lighting variations when compared to traditional frame-based cameras.

A brief description of powerlink industrial network is included in Section 2. The used materials and their interconnections are described in Section 3. Section 4 details the event-based FPGA network node, including camera communication, event-based processing and powerlink protocol implementation. Experimental tests and their results are provided in Section 5 and, finally, Section 6 provides conclusions.

## 2. POWERLINK IEEE 61158 Industrial Protocol

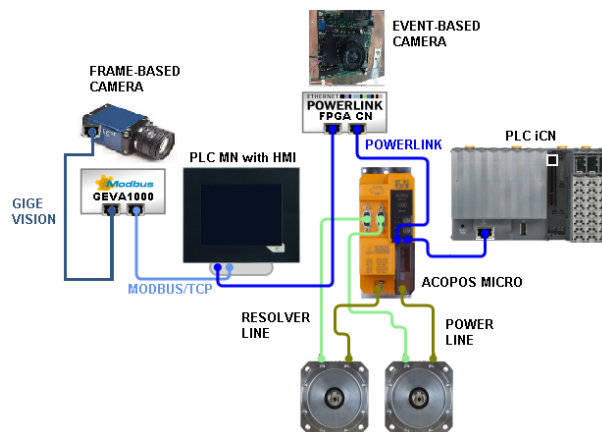
Ethernet POWERLINK is a protocol for communication among multiple industrial devices, machines and equipment [36]. It is designed with the goal of being implemented from the machine level to the process level, always involving communications in industrial plants. At the machine level, a high speed response is required, while, at the process level, efficiency in the transmission of a large amount of data is required. Some examples are: the sending of a setpoint position in a servomotor or reporting the state of a machine or complete automation system to a central supervision desk. Due to its deterministic approach, powerlink can be used in applications that require a high speed of data transfer in very short cycles such as machine vision and motion control machinery. POWERLINK is based on seven layers defined by the ISO/OSI model, like many other protocols. However, powerlink uses a different communication strategy and is based on slight modifications in the layers of the model according to the needs of speed and the amount of data to be transferred. For this reason, we have conducted a series of tests in order to characterize the advantages and disadvantages offered by this protocol [37], showing its feasibility. Powerlink is an object-oriented protocol based on CANopen protocol [38]; it uses a modified master/slave model where slaves can also share information among them. The master is referred to as Managing Node (MN) and the slave as the Controlled Node (CN). Powerlink is suited for a machine or a decentralized plant structure providing the users increased flexibility for adaptations and extensions due to its full adherence to the Ethernet standard IEEE 802.3, which yields two key features for its use in decentralized environments: cross-traffic and a free choice of network topology. The protocol application layer is defined as a carrier of all CANopen mechanisms [39] but also allows the use of other communication profiles which are out of scope in this work.

## 3. Materials and Methods

The proposed experimental setup was conceived to verify the two main novelties developed in this work: first, the behaviour of event-based and tracking algorithms and their advantages of implementation in an FPGA, including powerlink communication; and second, the capabilities of powerlink to provide enough speed when the Managing Node (MN) is receiving data from the FPGA node and, at the same time, transmitting data to a two-axis motor controller and a distributed I/O system as in a real industrial scenario. Additionally, the MN is exchanging data via MODBUS/TCP with a PC where frame-based camera computations are done, through a second communication link, all simultaneously. Thus, the whole system serves to compare two vision technologies: traditional industrial frame-based cameras versus event-based cameras. A complete industrial system was built, including an industrial cabinet with two emergency stops (one located in the cabinet and the second located near the positioning robot) meeting industrial safety regulations. Additionally, electrical protections, power supplies and wiring have been done to meet industrial standards. The block diagram describing the equipment and interconnections is shown in Figure 1. The following materials were used:

- A B&R (Eggelsberg, Austria) controller with Human Machine Interface (PLC MN/HMI), Power Panel C70, 5.7", 1 Link interface, 1 POWERLINK interface, 1 Ethernet interface 10BASE-T/100BASE-TX and 2 USB. This panel serves as the Managing Node (MN) in the POWERLINK network and slave in the MODBUS/TCP network. It also provides user visualization and parameter selection [40], together with the inverse kinematics computations for the two-axis robot movement.
- A B&R (Eggelsberg, Austria) PLC X20CP1382 as intelligent Controlled Node (PLC iCN) for distributed I/O control. Includes 14 digital inputs, 4 digital outputs, 4 digital inputs/outputs, 2 analog inputs, 2 USB, 1 RS232, 1 CAN bus, 1 POWERLINK, 1 Ethernet 10/100 Base-T [41].
- A B&R (Eggelsberg, Austria) two-axis ACOPOS micro 100PD.022 inverter module for servo motor control, POWERLINK interface, 2x resolver, 2 motor connections, 24 VDC [42].

- Two B&R (Eggenberg, Austria) Synchronous motors, self-cooling, with nominal speed of 3000 rpm for the robot movement [43].
- An Anybus HMS (Hamstad, Sweden) CompactCom M40 POWERLINK gateway [44] providing the physical POWERLINK connection to the FPGA.
- A Xilinx (Phoenix, AZ, USA) FPGA Zedboard [45] for event-based camera reading, event filtering algorithm computation and POWERLINK communication. The Xilinx board acts as a Controlled Node (CN) in the network.
- A non-commercial event-based CMOS camera with Selective Change Driven (SCD) vision, capable to produce a new event every  $1.7 \mu\text{s}$ , equivalent to 500 kfps in a conventional frame-based camera [46].
- An industrial PC Teledyne DALSA (Waterloo, Canada) GEVA1000 (2.4 GHz Dual Core, GigE x2, RS232, USB, 8 digital inputs 8 digital outputs) [47] connected via gigabit Ethernet with a frame-based camera GENIE M640 (CR-GEN3-C6400),  $1/3''$  format Charge-Coupled Device (CCD) with a resolution of  $640 \times 480$  operating at 64 frames per second at full resolution [48] and connected to the managing node controller via MODBUS/TCP.
- A ball movement system including a lighting control (on/off) and eight direct operated 2 port solenoid valves (SMC model VX21, Tokyo, Japan) [49] connected to the distributed I/O. The lighting consists of a white led stripe around the table where the ball is moving, controlled by a digital output from the distributed I/O node.



**Figure 1.** General view of the proposed system. Three network controlled nodes: Field Programmable Gate Array (FPGA) for event-based camera, Programmable Logic Controller (PLC) for distributed I/O and a two-axis servo controller, one managing node (Power Panel controller) and one MODBUS/TCP node (PC) for the frame-based camera.

### 3.1. Managing Node: Controller and HMI

The B&R Power Panel C70 controller receives data from the GEVA1000 PC via MODBUS/TCP using a generic Ethernet port. It also exchanges information with the powerlink controlled nodes through its specific POWERLINK interface port. The controller includes a 5.7" touch panel serving as user interface (HMI) for data visualization and parameter configuration. The controller computes the inverse kinematics equations (described below) for the robot positioning and sends the target position to the two-axis motor drive (Acospos micro) to position the robot according to the control option: frame-based or event-based.

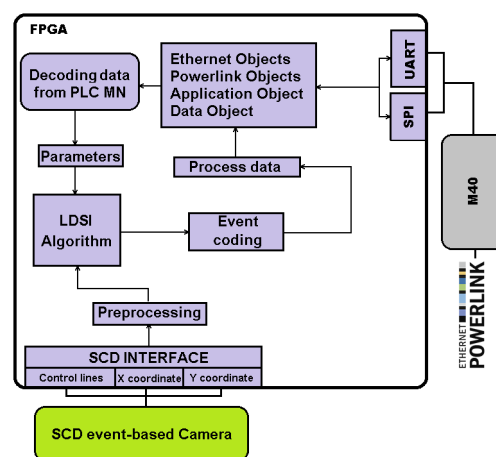
### 3.2. Event-Based Node: SCD Camera, FPGA and Powerlink Protocol

Unlike frame-based cameras, event-based cameras do not provide an industrial communication interface since they still are in an initial stage of development. Event-based cameras do not transmit a full data frame of the sensor size periodically; they only transmit information from those intensity



changing pixels. Thus, in a static scene, no information is generated; moving objects cause light changes in certain pixels and only those generate data. This approach results in a low data transfer rate and accurate positioning of pixels in the sensor array, and, as the intensity change is differential, a strong immunity to light exists. However, the information received from the changing pixels usually contains multiple spurious active pixels due to different noise sources from ambient conditions. This can be solved with a pre-processing algorithm eliminating the pixel activity out of the main scene, thus generating less data for the post-processing algorithm as object tracking, in this case.

The complete FPGA node is shown in Figure 2. The FPGA controls and receives data from the SCD event-based camera using its specific electronic interface [46,50]. The specific parallel port informs the FPGA the pixel address  $xy$  where the event has been generated. Once event data are received by the FPGA, noise removal, event filtering and object tracking algorithms are computed so that the object position coordinates are obtained. Then, the FPGA is executing the communication protocol so that object position is transmitted via powerlink communication to the Managing Node (PLC MN/HMI). This is done by connecting the FPGA with the Anybus CompactCom powerlink device which provides the physical powerlink bus connection. The FPGA is also receiving data from the PLC MN/HMI, which sends the parameter configuration data to the FPGA for the filtering and tracking algorithms. Furthermore, one of the most important tasks was the FPGA programming of all communication objects required to perform an standard powerlink data exchange among nodes. Despite this being transparent to the final working system, its optimization in programming is important so that the FPGA can meet the timing requirements specified by the standard.



**Figure 2.** Internal software developed for the FPGA controlled node (CN). The FPGA reads visual event-data from the Selective Change Driven (SCD) camera through a parallel custom signal port, computes the algorithms for filtering and object tracking, and exchanges information with the powerlink bus through the M40 Anybus device for physical bus connection with the PLC MN/HMI.

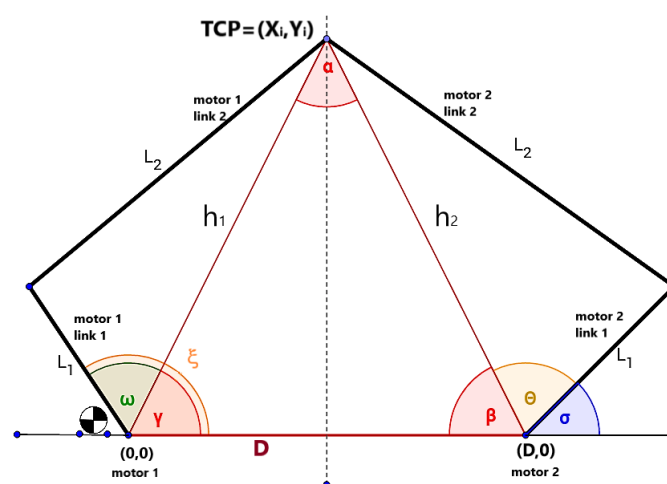
### 3.3. Frame-Based Industrial Vision System

The GEVA1000 industrial PC receives images from the frame-based camera, it also includes Sherlock commercial software (v6.0, Teledyne DALSA, Ontario, Canada) [51] for image processing used to track the object. The industrial PC and the HMI are connected by a MODBUS/TCP network where the master node is the industrial PC and the HMI acts as slave in the MODBUS/TCP network. Tracking algorithm configuration parameters can be adjusted from the HMI user interface so that the user can tune the algorithm. In addition to parameter configuration, the object position can be observed in the HMI for informative purposes despite a delay being observed between the current object position and that shown in the HMI, caused by the dynamic visualization option of the HMI. However, the object position is received in the PC with a minimum delay. The immediate availability of the object position allows the robot positioning algorithm to be calculated and position the robot with minimum delay, in case the user selects the option of positioning the robot using the frame-based camera.

Once the image is captured in the RGB color space, it is converted to gray scale with the usual 8-bit resolution. The space or region of interest to be analyzed is defined in this grayscale image. This region of the image will be binarized, going from grayscale to black and white. The conversion is done by means of the algorithm commonly known as 'threshold', i.e., if the pixel value is greater than a threshold defined by the user, the pixel becomes white, otherwise, it assigned to black. Next, an 'erosion' of the binarized result is done to eliminate noise events surrounding the object, converting into white those black pixels in the border of black blobs in the image. Finally, the 'area' algorithm is used to find objects represented as a black blobs on a white background. The blob position (matching the moving object) is obtained and converted into robot axis coordinates [52], which are sent to the controller for robot movement. The HMI includes a configuration screen where all algorithm parameters can be adjusted by the user: threshold level (in the range from 0 to 255) and area of interest (size in total pixel area).

### 3.4. Two-Axis Robot

The proposed robot is designed to position the Tool Center Point (TCP) in any position  $(X_i, Y_i)$  of a plane in a certain range. The robot is made of two servo motors separated a distance  $D$ , the first motor (first axis) is located in the origin  $(X, Y) = (0, 0)$  and the second motor is located in  $(X, Y) = (D, 0)$ . Each axis contains one joint, i.e., two links with length  $L_1$  and  $L_2$ , respectively, being equal on both axes. The final link in first and second axes are connected in a joint, forming the TCP. Figure 3 shows the scheme and links connection of the proposed robot system.



**Figure 3.** Two-axis robot design. Distances and angles for inverse kinematics calculation to obtain required angles for motor1 and motor 2 rotation to position TCP in a certain position in the plane  $(X_i, Y_i)$ .

According to Figure 3, the position of a certain  $(X_i, Y_i)$  point is separated a distance  $h_1$  and  $h_2$  in a straight line from the first and second motor, with an angle  $\beta$  and  $\gamma$ , respectively. The rotation angles  $\xi$  and  $\sigma$  of the first and second motor respectively are computed by the Equation (1). The obtained solution allows for positioning the TCP in  $(X_i, Y_i)$ :

$$\begin{aligned}
 h_1^2 &= X_i^2 + Y_i^2, \\
 h_2^2 &= (D - X_i)^2 + Y_i^2, \\
 \gamma &= \arccos\left(\frac{X_i}{h_1}\right), \\
 \beta &= \arccos\left(\frac{D - X_i}{h_2}\right), \\
 \omega &= \arccos\left(\frac{h_1^2 + L_1^2 - L_2^2}{2h_1L_1}\right), \\
 \Theta &= \arccos\left(\frac{h_2^2 + L_1^2 - L_2^2}{2h_1L_1}\right), \\
 \sigma &= 180 - \Theta - \beta, \\
 \xi &= \omega + \gamma.
 \end{aligned} \tag{1}$$

The robot arms were mechanically designed and created specifically for this application using Computer Aided Manufacturing (CAM-milled) aluminium for rigidity. The bearings are installed in the joints to reduce friction and improve accuracy.

The PLC MN/HMI node receives the estimated position from both event-based and frame-based systems. The user may choose the vision system controlling the robot. Then, the PLC MN/HMI computes the inverse kinematics described in Equation (1) and the rotation angles obtained are sent to the Acopos Micro servo driver.

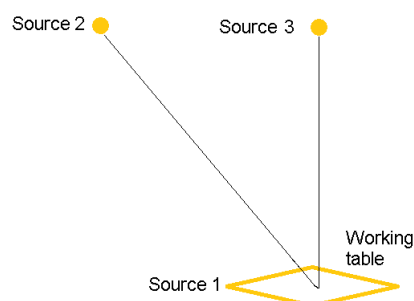
### 3.5. Pneumatic and Lighting System

The B&R PLC X20CP1382 serving as distributed I/O controlled node in the powerlink network controls the pneumatic system for air blowing to the object (a ball in this case). One additional output is used for LED lighting of the ball table (source 1).

By means of the PLC MN/HMI, the user can select different options for valve switching, changing speed and the sequence of activation so that different ball movements can be forced.

In order to test the effect of lighting conditions, three different sources of light were mounted as shown in Figure 4. With the combination of these sources, it is possible to obtain three different levels of light intensity over the working table. The combinations were defined as follows:

- **Lighting 1:** Source 1 on, sources 2 and 3 off.
- **Lighting 2:** Sources 1 and 2 on, source 3 off.
- **Lighting 3:** All sources on.



**Figure 4.** Scheme of placement of the lighting sources, source 1 is a constant background light, source 2 is placed so that it strikes at 45 degrees and source 3 strikes at 90 degrees on the table where the object is moving.

Light source 1 is always on to maintain a constant background light. Then, perpendicular (source 3) and oblique (source 2) lights are combined. Oblique light is created to generate lateral

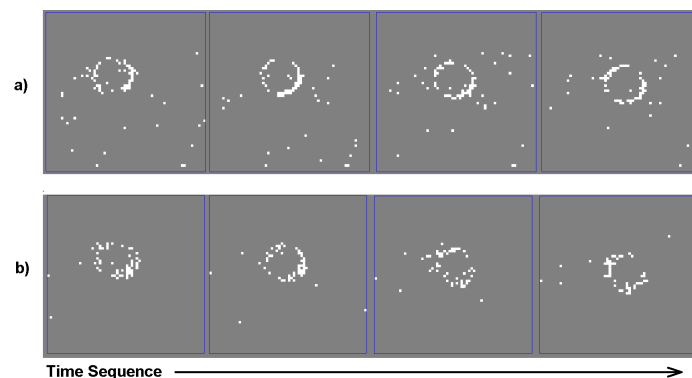


shadows so that robustness to detection can be verified. Lighting conditions recreate the normal ambient in industrial environments in image acquisition systems without a lighting box, i.e., specific close lighting (source 1) complemented by additional medium- and far- light sources (sources 2 and 3).

#### 4. Event Processing in the FPGA: Spurious Reduction, Filtering and Tracking

In this work, event processing is done in the FPGA for fast processing and reduction in data transmission. Two main algorithms were applied. First, those isolated spurious pixels generated by the event-based camera were eliminated, second, a bio-plausible event filter called LDSI proposed by the authors was applied [27], clearing the image and reducing the amount of redundant or irrelevant data. The jAER software (v1.7.1, INivation AG, Zurich, Switzerland) is used to visualize [53] results, a specialized application in Java for event-based devices. As jAER requires event data in Address Event Representation (AER) format, the FPGA included special functions converting event-data from the camera into AER data format. In recent years, AER has been used for interfacing chips containing multiple event receiving and/or event sending units (pixels, in case of a camera). In an AER link, an address that identifies the unit sending or receiving an event is transmitted on a bus [54,55].

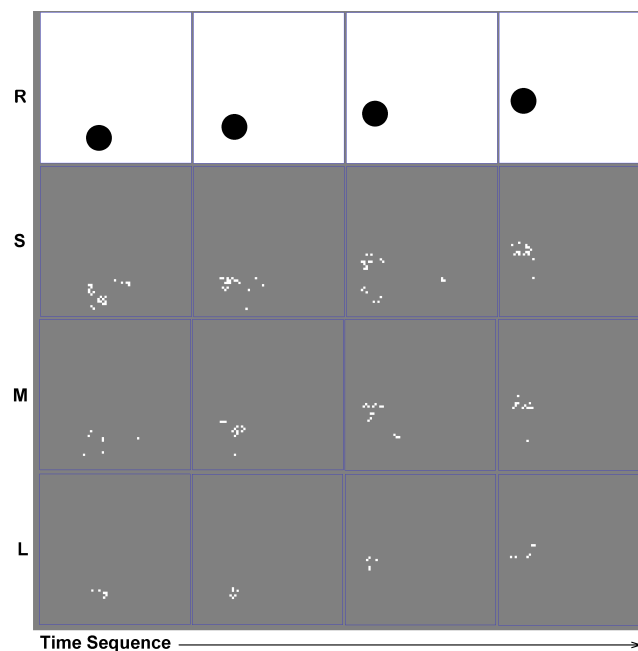
Figure 5 shows a circular object in the scene, in four different time instants. The object position (ball) can be appreciated as the circular set of white events. In Figure 5a, a large amount of spurious events can be seen. Spurious events are mainly generated by fluctuating light sources and by an effect of the intrinsic technology of these cameras [56]. The elimination of these events is the initial processing step of event data received by the FPGA. First, the FPGA learns which pixels in the camera report activity when there is a background in the static scene (in this case, no pixel should be active). Second, the FPGA saves these positions and discards them for subsequent processes since they are not providing useful information about the scene. This spurious pixel detection includes a limit in the number of discarded pixels (maximum, 20% of total pixels in the camera) to avoid a high loss of resolution. Results after spurious removal can be observed in Figure 5b; the same object is observed but spurious pixels are eliminated.



**Figure 5.** Spurious filter result in four different lapses of time. The circular object position can be appreciated as the white events forming a circle. (a) raw data generated from the event-based camera; (b) data after the spurious event removal.

After the elimination of spurious pixels, the LDSI algorithm is applied. This algorithm allows different configurations through parameters. Despite each parameter being able to be individually adjusted, three general configurations are defined to provide low, medium and high filtering level. According to the requirements of the application, it must be stronger or weaker in its filtering level. The model defined in this algorithm evaluates the event activity in a certain pixel, together with its neighbour pixels, and the event activity along time. Thus, those events more distant in time and space to others increase the possibility of being eliminated unless a new event is received in the same location inside a certain elapsed time.

The filter is bio-inspired, based on interconnected units working as neurons. Each neuron accounts for a potential level according to the events received from different inputs, also considering the elapsed time between events. This potential can be associated with the membrane potential in biological neurons. If the potential goes above a certain threshold, an output event is generated. The filter is based on two layers where each layer is composed of  $M \times N$  neurons where  $M \times N$  is the size of the pixel array in the event-based camera. After processing all layers, the filter output is an  $M \times N$  array. Doing this, the filter output maintains the same format and characteristics as the original data generated from the camera, which makes it 'transparent' for further processing modules. Figure 6 shows the results of the filter for low, medium and high filtering levels according to different parameter values, so called S, M and L, respectively. This figure shows the importance of the filter as the relevant information is kept; in this case, the position of the object, at a very low data production, benefiting at the same time with low processing burden and thus reducing data transfer in bus transmission.



**Figure 6.** Low (S), medium (M) and high (L) LDSI filtering. Row R shows the real position of the ball for each time instant analysed. Filter parameters allow the adjustment of output results. In this case, level S filtering values provide a clear location of the moving ball but more data to process. On the contrary, level L filtering provides the position with very low data production. It is clearly shown that relevant information is kept, at a very low data encoding (low number of events remaining).

The model defines a single neuron composed of two units associated with the nucleus (**Dlayer**) and the axon or synaptic terminals **Alayer**, being  $M \times N$  units in size. These units are arranged in two layers forming a neuronal-like structure. Each layer is defined by a bidimensional matrix of units identified by its  $xy$  coordinates in the matrix. Each unit in **Dlayer** and **Alayer** receive events from the input layer **Slayer** and modifies its internal potential value. A unit  $D_{xy}$  in **Dlayer** receives input events from the same  $xy$  position in the event generation layer (e.g., a sensory layer in an event sensor, or the output of a preceding layer). Then, the unit modifies its internal potential  $\vartheta_D(x, y)$ , which can be associated to the potential of the nucleus in a biological cell. Simultaneously, the units in **Alayer** modify its internal potential  $\vartheta_A(x, y)$  due to input events received in **Dlayer** units located in  $xy$ , and the vicinity. Each unit in a layer modifies its internal potential and, when potential in both **Dlayer** and **Alayer** is above a threshold, the unit in **Alayer** generates an output event, reflected in **Player**, which has the same structure and size as the input layer  $((M + 2) \times (N + 2))$ . This approach allows this LDSI filter to be included between already existing event processing modules since the

**Player** output can be interpreted as the original input layer. This approach is the same as in other processing areas where different filters may be added as pre-processing.

The LDSI filter defines the number of neighbour units from **Dlayer** affecting a unit  $A_{xy}$  in **Alayer**. This effect resembles a receptive field affecting potential in units nearby the generation of an event. The LDSI filter can be configured by the following parameters, related to the units in the layers:

- **Excitation level in Dlayer (ELD):** Magnitude of the potential that a unit in the  $xy$  unit of **Dlayer** increases when an event is received from the unit located in the same  $xy$  unit in **Slayer**.
- **Excitation level in Alayer (ELA):** The potential increment in the  $xy$  unit of **Alayer** due to an event in the same  $xy$  unit of **Dlayer**.
- **Excitation level in Alayer neighbouring units (ELAN):** When an event is produced in an  $xy$  unit of **Dlayer**, ELAN corresponds to the potential increment of units in **Alayer** the vicinity of the  $xy$  unit.
- **Threshold potential level in Dlayer (TPD):** Defines the minimum value of excitation required for a certain unit in **Dlayer** to generate an output event.
- **Threshold potential level in Alayer (TPA):** Defines the minimum value of excitation required for a certain unit in **Alayer** to generate an output event.
- **Decrement of potential in Dlayer (DPD):** The value of potential to be decremented in **Dlayer** once parameter MTR (Maximum Time to Remember) has elapsed.
- **Decrement of potential in Alayer (DPA):** The value of potential to be decremented in **Alayer** once parameter MTR (Maximum Time to Remember) has elapsed.

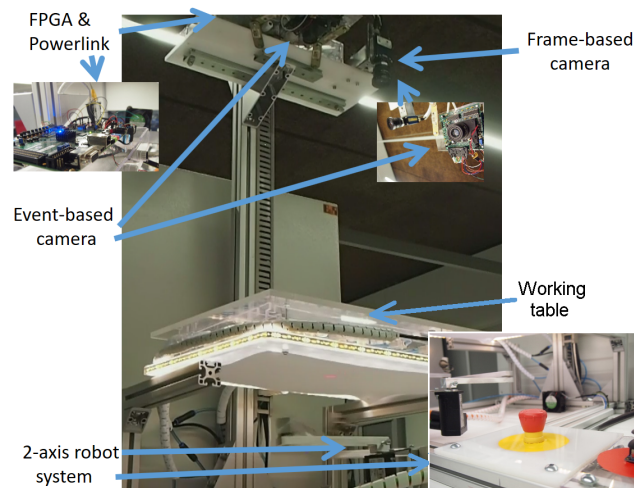
This set of parameters allows for tuning the output of events. Depending on the parameters, more strength can be given to events in the same  $xy$  position of the output, or provide more importance to the activation of events in a close field. The latter is the usual case for object detection as the events are not isolated. A proper configuration in this direction will remove noise in the form of isolated events and enhance the object borders. The filter can be adjusted from the null output event generation to a transparent mode where all input events are directly transferred into the output.

After processing, the event data from the LDSI filter, the output is computed to track the position of the moving object. A vicinity algorithm is applied so that the position of the event with most neighbouring events is the winner, the latest being the most valid in case of several positions having the same vicinity value. It is important to note so that the tracking provides a good performance, and the LDSI filter must be fine-tuned to avoid noise influence generating false event positions.

## 5. Test and Results

In Figure 7, the experimental setup for the test is shown. Both event-based and frame-based cameras are able to capture data from the scene simultaneously, and the user, through the HMI, can decide which vision system will control the robot for tracking the object in the working table. The robot position controller is able to receive data from the FPGA node and send it to the motor drive of the robot with enough speed to provide positioning when the event-based system is selected; alternatively, the position data goes from the camera to the industrial PC, later to the PLC MN/HMI, and finally to the robot motor drive when the frame-based system is selected. Both technologies are able to provide good positioning, also showing that bus communications respond with enough speed as to maintain a fast motor response when continuous and fast changes in position must be executed.

The filtering and conditioning of the events from the event-based camera were also tested. The test consisted on high speed movement across the working table at three different sets of filter parameters, under three different sources of light for both vision systems. All combinations were tested, i.e., S, M and L filter options for each camera combined with lighting 1, 2 and 3. In total, 27 tests were done. Tables 1 and 2 present the used values for the three set of parameters for event-based and frame-based cameras, respectively.



**Figure 7.** General view of the experimental setup for the tracking system. FPGA node for event-based camera on top, frame-based and event-based cameras, ball movement blowing system with lighting, two-axis robot and electrical cabinet including safety.

**Table 1.** LDSI filter parameters for the event-based system Selective Change Driven (SCD).

| Set | ELD | ELA | ELAN | TPD | TPA | DPD | DPA |
|-----|-----|-----|------|-----|-----|-----|-----|
| S   | 1   | 1   | 1    | 2   | 2   | 0   | 0   |
| M   | 3   | 4   | 4    | 6   | 6   | 1   | 1   |
| L   | 3   | 3   | 3    | 9   | 9   | 1   | 1   |

For the selection of the parameters of the LDSI filter, a configuration step is necessary to tune the parameters to the scene characteristics. It is recommended to start with higher values of TPA and TPD than ELD, ELA and ELAN; at least twice the value if zero noise and a few events for the moving objects in scene are desired, high filtering level (L). Then, for a less restrictive filter, values of ELD, ELA and ELAN must be decreased in order to obtain a better definition of the objects. In case the high noise level and the object is not sharp, then DPD and DPA values must be reduced. According to the filter structure described in Section 4, by following these steps, more importance can be given to isolated or grouped events, generating a different filter output, accordingly.

Concerning the parameters chosen for the three levels of detection in the frame-based system, namely S, M and L for small, medium and high filtering level, the most important value is the binarization threshold level (from 0 to 255), followed by the size of the area generated by the remaining pixels forming a blob after binarization. Given values in Table 2 were tuned according to the lighting situation in the proposed scenarios, and the type of object to be detected.

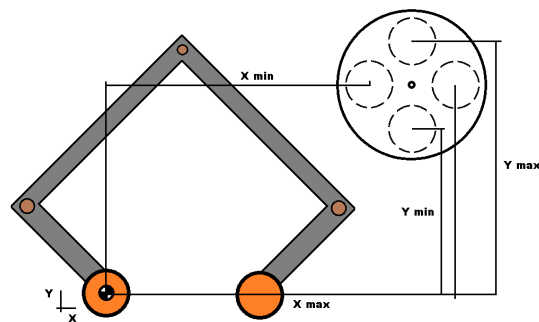
**Table 2.** Frame-based vision (GENIE) system parameters.

| Set | Threshold | Area |
|-----|-----------|------|
| S   | 25        | 500  |
| M   | 60        | 1000 |
| L   | 100       | inf  |

In this case, the tracking does not include prediction of future object position. For this reason, there exists a delay in the actual ball position and the position where the Tool Center Point of the robot (TCP) is moving, which is negligible in most of the cases. We have created a video repository [57] showing the performance of the robot tracking activity when following the object randomly impulsed by the pneumatic valves. This test shows in a visual form that the robot is capable of reaching high

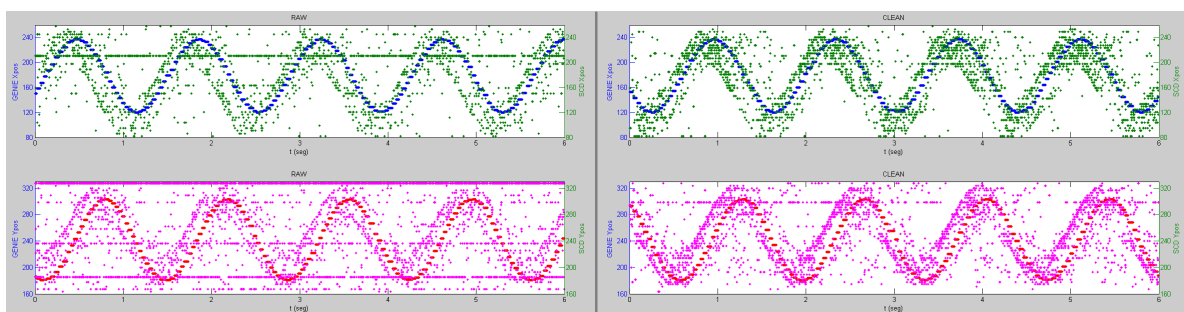
speed when following an object, in both vision systems, which provide a good response for such high demanding process.

Nevertheless, with the aim to verify the event-based camera performance and compare with the frame-based camera in a repeatable and quantitative test, a second experiment was performed. A white turning disc rotating at constant speed, with a black circle on it (easily identifiable by a vision system) was located under the field of vision of both systems. Figure 8 shows the situation of the rotating disc and distance to the origin of coordinates for the robot. The figure also shows the distances that the vision systems must report for the object located in the turning disc ( $X_{max}, X_{min}, Y_{max}, Y_{min}$ ). The main goal in this case is to detect the object constantly moving in a  $XY$  plane, which will produce a sinusoidal pattern in both axes, with a sinusoidal amplitude of  $|X_{max} - X_{min}|$  and  $|Y_{max} - Y_{min}|$ , respectively. Doing this, it was possible to detect and measure the difference in time of object position location reported from both vision systems, and the difference in position detection.



**Figure 8.** Diagram of the coordinates and situation of a rotating disc with a circular object on it. This setup is used to obtain a predictable movement in a known  $XY$  plane and compare camera results.

Figure 9 shows the generated events when data are directly received from the event-based camera with no processing (left), and the resulting event-data when the spurious events are removed (right). In blue and red, the position obtained from the frame-based camera in the  $X$  and  $Y$  axes is presented, respectively. In green and pink, the position obtained from the event-based camera in the  $X$  and  $Y$  axes is shown, respectively. For the left graph, the straight lines caused by the spurious pixels can be easily appreciated. This is due to events from pixel positions generating events continuously. In addition, Figure 9 shows another effect: discarding spurious data reduces the events and thus the computation time for the FPGA. The previous elimination of spurious events permits the FPGA to process useful data and reduce the computation time for them. This effect is appreciated in the figure as, apart from highly reducing the horizontal lines, the remaining events are more concentrated in the area of the sinusoid shape (right) in contrast to more scattered events when no elimination is done (left).



**Figure 9.** (Left) raw event data without event processing generated by the event-based camera ( $X$ -axis in green,  $Y$ -axis in pink) and the frame-based camera ( $X$ -axis in blue,  $Y$ -axis in red); (Right) event data after spurious events being discarded. For the event-based camera, the constant *dead* pixels appearing as horizontal lines are eliminated and remaining events are less scattered. There are more events close to the main sinusoidal signal as the time to process a spurious event is now used to process a real event.

A numerical estimation of the difference in time reported by each camera, for the same position, was measured. The  $\Delta t$  between the timestamp of each camera when the position was received by the controller was measured. Table 3 shows the summary of the results for all tested combinations using three sets of parameters for each vision system as described earlier. Positive values mean less time to report the same position for the event-based system with respect to the frame-based system and negative values, the opposite. Equation (2) shows how values are computed for the X-axis, the same arises for the Y-axis. As seen in the equation, the time value where the local maximum or minimum appears for the event-based and the frame-based cameras is obtained, the difference being the resulting values shown in Table 3. Despite a high data dispersion being observed, the event-based camera is able to provide the position with an average of 98.01 ms ahead of the frame-based camera. The  $\Delta t$  can reach up to 333.63 ms time advance, although some delay can also be observed in some cases (up to  $-276.02$  ms):

$$\begin{aligned}\Delta t_{max} &= X_{max}(event) - X_{max}(frame), \\ \Delta t_{min} &= X_{min}(event) - X_{min}(frame).\end{aligned}\quad (2)$$

**Table 3.** Comparative summary of time difference  $\Delta t$  between the event-based and the frame-based vision systems when reporting the same position to the controller.

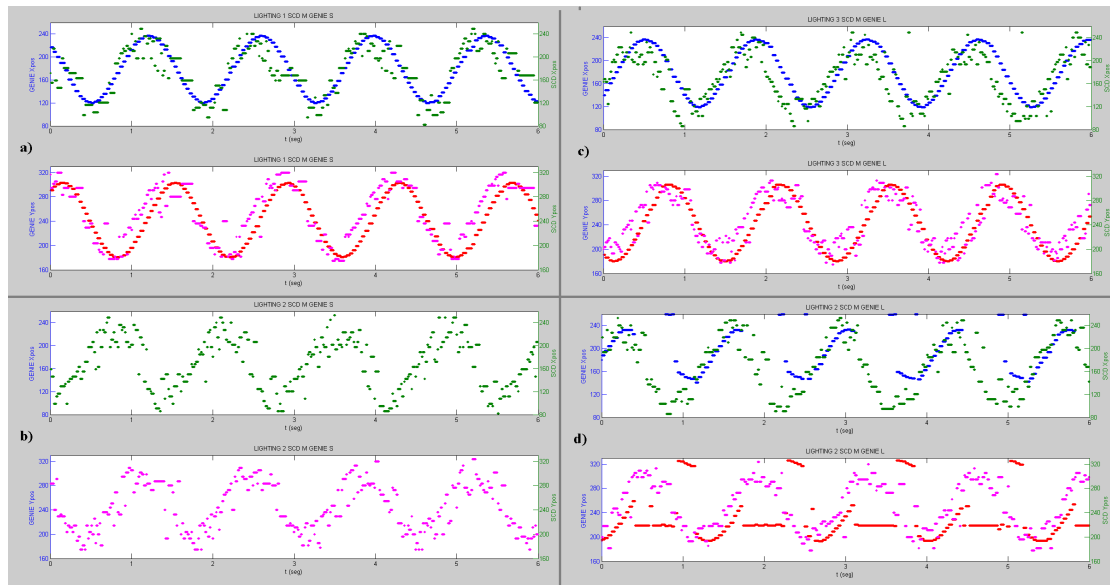
|                     | S         | M         | L         | Global    |
|---------------------|-----------|-----------|-----------|-----------|
| <b>Average</b> (ms) | 109.67    | 99.63     | 88.39     | 98.01     |
| <b>Std Dev</b> (ms) | 98.05     | 90.34     | 112.4931  | 97.72     |
| <b>Maximum</b> (ms) | 331.11    | 300.12    | 333.63    | 333.63    |
| <b>Minimum</b> (ms) | $-136.80$ | $-177.56$ | $-276.02$ | $-276.02$ |

The response time is important, but also robustness to changes in light conditions. Three different scenarios were implemented using the lighting sets described in Section 3.5. Figure 10 shows the effect of different filtering options (S, M or L) under three lighting scenarios. In all cases, for the event-based camera, it can be clearly seen that only relevant events are kept after filtering, in comparison with all events received from the camera (Figure 9), which shows the feasibility of tracking position at a reduced number of data.

Figure 10 shows how the frame-based system can track the position under stable lighting conditions (lighting 1 and 3 in Figure 10a,c) but not in case of oblique light as for lighting 2 shown in Figure 10b,d. Thus, the event-based camera is more robust to light changes, especially when light strikes obliquely. For this reason, many frame-based systems require specific tunnels where inspected objects go through; however, many applications do not allow its use and a poor result of frame-based cameras is obtained. It is also interesting to observe that, under lighting 2 (oblique light), the effects of shadows cause some double detection in the event-based system, i.e., both object borders are detected. This effect can be clearly seen in Figure 10b.

Table 4 summarizes the moving object position detected at the extreme values as an average in all performed experiments. The values are referenced to the working coordinates of the robot as depicted in Figure 8. The standard deviation in the GENIE is higher than the event-based SCD showing that values may vary in a wide range, mainly due to lighting. For the event-based, the position is kept in a close range (lower standard deviation) for all performed tests. This test proves that the event-based SCD camera is more robust to light changes, mainly due to its differential working principle, i.e., event are created when light intensity difference exists, regardless of the value. Concerning the average value, it is interesting to see how the position value changes depending on the type of camera. For the Y-axis, the difference is about 8 mm, while in the X-axis there is a difference of 28.9 mm and 15.9 mm for the minimum and the maximum, respectively. The main reason for this difference lies in the tracking algorithm: while SCD tends to detect the object in the extreme border, the GENIE extracts a centered position. In this case, the effect is more appreciated in the X-axis due to the lighting position, which creates wider shadows in this axis direction.





**Figure 10.** Comparison of the frame-based (GENIE) and event-based (SCD) vision system for different lighting conditions and filtering options. (a) lighting 1, parameters M and S for SCD and GENIE, respectively; (b) lighting 3, parameters M and L for SCD and GENIE, respectively; (c) lighting 2, parameters M and S for SCD and GENIE, respectively; (d) lighting 2, parameters M and L for SCD and GENIE, respectively. Both systems show a good behaviour in (a) and (c), with lighting 1 and 3. However, using lighting 2 and the same parameters as in (a) and (c), the frame-based vision system (GENIE) cannot track the object totally as shown in (b), or partially, as in (d). The event-based camera keeps reporting the position despite the used lighting.

From the obtained data, two main results are obtained. First, the SCD camera is more robust under changes in lighting conditions. Second, the speed in detection is faster when compared to the frame-based, being able to detect the object changes in about 100 ms earlier.

**Table 4.** Position estimation for frame-based and event-based systems. Values correspond to average and standard deviation in the X-axis and Y-axis distance from the robot reference coordinates, in millimeters, as shown in Figure 8.

|              | Minimum (mm)      | Maximum (mm)       |
|--------------|-------------------|--------------------|
| GENIE Y-axis | $182.04 \pm 3.93$ | $309.36 \pm 17.74$ |
| SCD Y-axis   | $174.68 \pm 1.23$ | $317.71 \pm 4.96$  |
| GENIE X-axis | $119.43 \pm 7.60$ | $240.61 \pm 8.26$  |
| SCD X-axis   | $90.48 \pm 3.22$  | $256.51 \pm 2.18$  |

## 6. Conclusions

One of the main differences between event-based and frame-based image sensing techniques lies in the fact that frame-based cameras require a high computational cost: a dedicated industrial PC with isolated Ethernet communication between PC and frame-based camera is required, image processing algorithms are computationally demanding, and power consumption is high. However, for certain applications, the same result can be achieved by an event-based system at a low computational burden, being implemented in a low cost device such as microcontrollers or simple FPGA. In this case, an FPGA is used.

Robustness to light changes and response time are the main advantages of event-based cameras when compared to frame-based cameras. However, event-data processing is required. Removing spurious events and applying event filtering generate less events (impacting in data reduction,

advantageous in case of further data transmission or processing) and those generated events are more accurate, closer to the scene of interest. This is of special interest in high speed object detection where time response is essential. In this case, the event-based camera itself, together with low computation required, can provide a solution. This work includes the following novelties: event-based processing, only leaving relevant events, which, in turn, reduces data for further transmission (lower bandwidth use); comparison of response time and lighting conditions robustness with frame-based cameras by means of a two-axis robot; integration of an event-based camera into the powerlink IEEE 61158 industrial communication as a network node.

On the other hand, the integration of new devices in standard industrial communication protocols is a need in many factories where custom processes need to be carried out, and many modern tasks are requiring high-throughput communication interfaces. Having smart network nodes able to provide relevant information using reduced bandwidth is a must. This work describes the successful integration of an event-based camera into the IEEE standard powerlink communication bus by means of an FPGA. The camera is seen as a controlled node in the network where a commercial B&R controller acts as managing node. In addition, the FPGA performs local computing for data filtering and object tracking, sending the tracking position to the controlled node which in turn, controls a two-axis robot custom designed where the inverse kinematics computation for positioning is done by the managing node. Alternatively, the two-axis robot can be controlled by a traditional frame-based camera so that a comparison between both image acquisition and tracking processing was done. The frame-based camera requires the use of a dedicated computer with isolated high bandwidth communication between the PC and the camera, resulting in a high power consumption and a more complex system while the same goal can be achieved by a simpler system, which makes the event-based technology even more interesting to spread in the industrial field with new applications.

To summarize, this work proposes the integration of an event based sensor into the standard Ethernet powerlink industrial network by developing a network node based on an FPGA. The FPGA node includes the event-data acquisition from the camera, event filtering and tracking algorithms, and protocol stack for data communications. An experimental setup for a high demanding task is developed so that the advantages of event-based cameras versus frame-based cameras are demonstrated.

**Author Contributions:** Conceptualization, A.R.-M.; Data Curation, L.M.; Formal Analysis, J.B.-A.; Investigation, J.B.-A.; Methodology, A.R.-M.; Project Administration, A.R.-M.; Resources, T.I.; Software, J.B.-A.; Supervision, A.R.-M.; Validation, J.B.-A.; Writing—Original Draft, J.B.-A. and J.S.; Writing—Review and Editing, A.R.-M. and T.I.

**Funding:** This research received the funds obtained as compensation for the first prize in EPL awards 2017 given by the Ethernet Powerlink Standardization Group. <https://www.ethernet-powerlink.org/>.

**Acknowledgments:** The authors would like to thank Fernando Pardo for his help and support in communicating with the SCD camera developed by his research group at the University of Valencia, and Vicente Martinez for his support in the mechanical design and manufacturing of the robot links. We would also like to thank Alejandro Cortina for his collaboration in setting up the entire electrical system and cabinet design.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Decotignie, J.D. Ethernet-Based Real-Time and Industrial Communications. *Proc. IEEE* **2005**, *93*, 1102–1117. [[CrossRef](#)]
2. Berner, R.; Brandli, C.; Yang, M.; Liu, S.C.; Delbruck, T. A  $240 \times 180$  10 mW 12  $\mu$ s latency sparse-output vision sensor for mobile applications. In *2013 Symposium on VLSI Circuits (VLSIC)*; IEEE: Piscataway, NJ, USA, 2013; pp. C186–C187.
3. Izhikevich, E.M. Simple model of spiking neurons. *IEEE Trans. Neural Netw.* **2003**, *14*, 1569–1572. [[CrossRef](#)] [[PubMed](#)]
4. Furber, S.B.; Lester, D.R.; Plana, L.A.; Garside, J.D.; Painkras, E.; Temple, S.; Brown, A.D. Overview of the SpiNNaker System Architecture. *IEEE Trans. Comput.* **2013**, *62*, 2454–2467. [[CrossRef](#)]

5. Liu, S.C.; Delbruck, T.; Indiveri, G.; Whatley, A.; Douglas, R. *Event-Based Neuromorphic Systems*; Wiley: Hoboken, NJ, USA, 2014.
6. Rios-Navarro, A.; Cerezuela-Escudero, E.; Dominguez-Morales, M.; Jimenez-Fernandez, A.; Jimenez-Moreno, G.; Linares-Barranco, A. Real-time motor rotation frequency detection with event-based visual and spike-based auditory AER sensory integration for FPGA. In Proceedings of the 2015 International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP), Krakow, Poland, 17–19 June 2015; pp. 1–6.
7. Serrano-Gotarredona, R.; Serrano-Gotarredona, T.; Acosta-Jimenez, A.J.; Linares-Barranco, B. An arbitrary kernel convolution AER-transceiver chip for real-time image filtering. In Proceedings of the 2006 IEEE International Symposium on Circuits and Systems, Island of Kos, Greece, 21–24 May 2006; p. 4.
8. Rivas-Perez, M.; Linares-Barranco, A.; Jimenez-Fernandez, A.; Civit, A.; Jimenez, G. AER spike-processing filter simulator: Implementation of an AER simulator based on cellular automata. In Proceedings of the International Conference on Signal Processing and Multimedia Applications, Seville, Spain, 18–21 July 2011; pp. 1–6.
9. Espínola, A.; Romay, A.; Baidyk, T.; Kussul, E. Robust vision system to illumination changes in a color-dependent task. In Proceedings of the 2011 IEEE International Conference on Robotics and Biomimetics, Phuket, Thailand, 7–11 December 2011; pp. 521–526.
10. Lin, W.K.; Uang, C.M.; Wang, P.C.; Ho, Z.S. LED strobe lighting for machine vision inspection. In Proceedings of the 2013 International Symposium on Next-Generation Electronics, Kaohsiung, Taiwan, 25–26 February 2013; pp. 345–346.
11. Kim, H.; Cho, K.; Kim, S.; Kim, J. Color mixing and random search for optimal illumination in machine vision. In Proceedings of the 2013 IEEE/SICE International Symposium on System Integration, Kobe, Japan, 15–17 December 2013; pp. 907–912.
12. Camuñas-Mesa, L.A.; Serrano-Gotarredona, T.; Linares-Barranco, B. Event-driven sensing and processing for high-speed robotic vision. In Proceedings of the 2014 IEEE Biomedical Circuits and Systems Conference (BioCAS), Lausanne, Switzerland, 22–24 October 2014; pp. 516–519.
13. Partzsch, J.; Mayr, C.; Vogginger, B.; Schüffny, R.; Rast, A.; Plana, L.; Furber, S. Live demonstration: Ethernet communication linking two large-scale neuromorphic systems. In Proceedings of the 2013 European Conference on Circuit Theory and Design (ECCTD), Dresden, Germany, 8–12 September 2013.
14. Fasnacht, D.B.; Whatley, A.M.; Indiveri, G. A serial communication infrastructure for multi-chip address event systems. In Proceedings of the 2008 IEEE International Symposium on Circuits and Systems, Seattle, WA, USA, 18–21 May 2008; pp. 648–651.
15. Miskowicz, M. Send-On-Delta Concept: An Event-Based Data Reporting Strategy. *Sensors* **2006**, *6*, 49–63. [[CrossRef](#)]
16. Diaz-Cacho, M.; Delgado, E.; Barreiro, A.; Falcon, P. Basic Send-on-Delta Sampling for Signal Tracking-Error Reduction. *Sensors* **2017**, *17*, 312. [[CrossRef](#)] [[PubMed](#)]
17. Socas, R.; Dormido, S.; Dormido, R.; Fabregas, E. Event-Based Control Strategy for Mobile Robots in Wireless Environments. *Sensors* **2015**, *15*, 30076–30092. [[CrossRef](#)] [[PubMed](#)]
18. Santos, C.; Martínez-Rey, M.; Espinosa, F.; Gardel, A.; Santiso, E. Event-Based Sensing and Control for Remote Robot Guidance: An Experimental Case. *Sensors* **2017**, *17*, 2034.
19. Acho, L. Event-Driven Observer-Based Smart-Sensors for Output Feedback Control of Linear Systems. *Sensors* **2017**, *17*, 2028. [[CrossRef](#)] [[PubMed](#)]
20. Sivilotti, M.A.; Mahowald, M.A.; Mead, C.A. Real-Time Visual Computations Using Analog CMOS Processing Arrays. *Adv. Res. VLSI Proc. 1987 Conf.* **1987**, *1*, 295–311.
21. Mead, C.A.; Mahowald, M. A silicon model of early visual processing. *Neural Netw.* **1988**, *1*, 91–97. [[CrossRef](#)]
22. Mead, C. Neuromorphic electronic systems. *Proc. IEEE* **1990**, *78*, 1629–1636. [[CrossRef](#)]
23. Camunas-Mesa, L.; Zamarreno-Ramos, C.; Linares-Barranco, A.; Acosta-Jimenez, A.J.; Serrano-Gotarredona, T.; Linares-Barranco, B. An Event-Driven Multi-Kernel Convolution Processor Module for Event-Driven Vision Sensors. *IEEE J. Solid-State Circuits* **2012**, *47*, 504–517. [[CrossRef](#)]
24. Zhao, B.; Ding, R.; Chen, S.; Linares-Barranco, B.; Tang, H. Feedforward Categorization on AER Motion Events Using Cortex-Like Features in a Spiking Neural Network. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 1963–1978. [[CrossRef](#)] [[PubMed](#)]

25. Jimenez-Fernandez, A.; del Bosh, J.L.F.; Paz-Vicente, R.; Linares-Barranco, A.; Jiménez, G. Neuro-inspired system for real-time vision sensor tilt correction. In Proceedings of the 2010 IEEE International Symposium on Circuits and Systems, Paris, France, 30 May–2 June 2010; pp. 1394–1397.
26. Miskowicz, M.E. *Event-Based Control and Signal Processing*; CRC Press: Boca Raton, FL, USA, 2015.
27. Barrios-Aviles, J.; Iakymchuk, T.; Samaniego, J.; Rosado-Muñoz, A. An Event-based Fast Movement Detection Algorithm for a Positioning Robot Using POWERLINK Communication. *CoRR* **2017**, arXiv:1707.07188.
28. ODVA. *Technology Overview Series: Ethernet/IP, CIP on Ethernet Technology*; Technical Report; ODVA Inc.: Ann Arbor, MI, USA, 2016.
29. Pigan, R.; Metter, M. *Automating with PROFINET: Industrial Communication Based on Industrial Ethernet*, 2nd ed.; Publicis MCD Verlag: Erlangen, Germany, 2008.
30. EtherCAT. *EtherCAT—The Ethernet Fieldbus*; Technical Report; EtherCAT Organisation: Nuremberg, Germany, 2014.
31. Ethernet Powerlink Standardisation Group (EPG). *Ethernet POWERLINK Communication Profile Specification*; Technical Report; EPG: Fredersdorf, Germany, 2013.
32. Farabet, C.; Paz, R.; Perez-Carrasco, J.; Zamarreno, C.; Linares-Barranco, A.; LeCun, Y.; Culurciello, E.; Serrano-Gotarredona, T.; Linares-Barranco, B. Comparison Between Frame-Constrained Fix-Pixel-Value and Frame-Free Spiking-Dynamic-Pixel ConvNets for Visual Processing. *Front. Neurosci.* **2012**, *6*, 32. [[CrossRef](#)] [[PubMed](#)]
33. Oliver, R.S.; Craciunas, S.S.; Stöger, G. Analysis of Deterministic Ethernet scheduling for the Industrial Internet of Things. In Proceedings of the 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Athens, Greece, 1–3 December 2014; pp. 320–324.
34. Yang, H.; Lin, H.; Li, J.; Tao, Y. The architecture and real-time communication of CNC systems based on switched Ethernet. In Proceedings of the 2010 2nd International Conference on Computer Engineering and Technology, Chengdu, China, 16–18 April 2010; Volume 1, pp. V1:169–V1:173.
35. Gong, Z.; Liu, B.; Yang, S.; Gui, X. Analysis of industrial ethernet's reliability and realtime performance. In Proceedings of the 2009 8th International Conference on Reliability, Maintainability and Safety, Chengdu, China, 20–24 July 2009; pp. 1133–1136.
36. IEEE Standard for Industrial Hard Real-Time Communication. *IEEE Std 61158-2017 (Adoption of EPG DS 301)*; IEEE: Piscataway, NJ, USA, 2017; pp. 1–395.
37. Barrios-Avilés, J.; Rosado-Muñoz, A.; Iakymchuk, T.; García-Chulbi, M. POWERLINK and Ethernet/IP Comparison as Robust Industrial Ethernet Protocols. *IFAC-PapersOnLine* **2017**, *50*, 363–368. [[CrossRef](#)]
38. Danielis, P.; Skodzik, J.; Altmann, V.; Schweissguth, E.B.; Golasowski, F.; Timmermann, D.; Schacht, J. Survey on real-time communication via ethernet in industrial automation environments. In Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA), Barcelona, Spain, 16–19 September 2014; pp. 1–8.
39. CANopen. *CANopen The Standardized Embedded Network*; Technical Report; CAN: Nuremberg, Germany, 2016.
40. B&R. *Power Panel C70 User's Manual v1.1*; Technical Report; B&R Industrial Automation: Eggelsberg, Austria, 2015.
41. B&R. *X20 System User's Manual v3.35*; Technical Report; B&R Industrial Automation: Eggelsberg, Austria, 2016.
42. B&R. *ACOPOSmicro User's Manual v1.20*; Technical Report; B&R Industrial Automation: Eggelsberg, Austria, 2016.
43. B&R. *8LVA Three-Phase Synchronous Motors. User's Manual v1.0*; Technical Report; B&R Industrial Automation: Eggelsberg, Austria, 2017.
44. ANYBUS. *CompactCom M40 Module—Powerlink. MMA316 Version 2. Network Guide, Hardware Design Guide and Software Design Guide*; Technical Report; HMS Industrial Networks: Hamstad, Sweden, 2017.
45. Avnet. *ZedBoard (Zynq Evaluation and Development). Hardware User's Guide v2.2*; Technical Report; Avnet: Phoenix, AZ, USA, 2014.
46. Pardo, F.; Boluda, J.A.; Vegara, F. Selective Change Driven Vision Sensor with Continuous-Time Logarithmic Photoreceptor and Winner-Take-All Circuit for Pixel Selection. *IEEE J. Solid-State Circuits* **2015**, *50*, 786–798. [[CrossRef](#)]

47. DALSA. *GV1000 Vision System. Installation Manual v4.5*; Technical Report; Teledyne DALSA: Waterloo, ON, Canada, 2016.
48. DALSA. *GENIE Monochrome Series Manual. CR-GEN3-M640X*; Technical Report; Teledyne DALSA: Waterloo, ON, Canada, 2011.
49. SMC. *VX21 Solenoid Valve Manual*; Technical Report; SMC Corporation: Tokyo, Japan, 2016.
50. Boluda, J.A.; Zuccarello, P.; Pardo, F.; Vegara, F. Selective Change Driven Imaging: A Biomimetic Visual Sensing Strategy. *Sensors* **2011**, *11*, 11000–11020. [[CrossRef](#)] [[PubMed](#)]
51. Teledyne. *Sherlock Vision System Software. User Manual and Embedded User Manual*; Technical Report; Teledyne DALSA: Waterloo, ON, Canada, 2017.
52. Dawson-Howe, K. *A Practical Introduction to Computer Vision with OpenCV*; Wiley and Sons Inc.: Hoboken, NJ, USA, 2014.
53. Open Source Software. User Guide jaER: Java Tools for AER Neuromorphic Processing. Available online: <https://inivation.com/support/software/jaer/> (accessed on 6 November 2018).
54. Mahowald, M. *An Analog VLSI System for Stereoscopic Vision*; Kluwer Academic Publishers: Norwell, MA, USA, 1994.
55. Lazzaro, J.; Wawrzynek, J.; Mahowald, M.; Sivilotti, M.; Gillespie, D. Silicon auditory processors as computer peripherals. *IEEE Trans. Neural Netw.* **1993**, *4*, 523–528. [[CrossRef](#)] [[PubMed](#)]
56. Pardo, F.; Boluda, J.A.; Vegara, F. Random telegraph signal transients in active logarithmic continuous-time vision sensors. *Solid-State Electron.* **2015**, *114*, 111–114. [[CrossRef](#)]
57. Barrios, J. SCD and GENIE Tracking Videos with a Two-Axis Robot. Available online: [https://www.youtube.com/playlist?list=PLLeYDuO4OSjTaARxjGDGcnRMGTTnJ3KN\\_](https://www.youtube.com/playlist?list=PLLeYDuO4OSjTaARxjGDGcnRMGTTnJ3KN_) (accessed on 6 November 2018).



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).