Event-based control system on FPGA applied to the pencil balancer robotic platform

F. Perez-Peña, T. Delbruck, S.C. Liu
Institute of NeuroInformatics, University of Zurich
ETHZ Zurich,Switzerland
Email: fernandoperez.pena@uca.es, {tobi,shih}@ini.uzh.ch

A. Linares-Barranco, G. Jimenez-Moreno
Computer Architecture and Technology, ETSI Informatica
University of Seville, Spain
Email: {alinares,gaji}@us.es

***Abstract*—An event-based motor controller design is presented. The system is designed to solve the classic inverted pendulum problem by using a robotic platform and a totally neuro-inspired event-based mechanism. Specifically, DVS retinas provide feedback and an FPGA implements control. The robotic platform used is the so called 'pencil balancer'. The retinas provide visual information to the FPGA that processes it and obtains the center of mass of the pencil. Once this center of mass is averaged over time, it is used joint with the cart position provided by a flat potentiometer bar to compute the angle of the pencil from the vertical. The angle is delivered to an event-based Proportional-Derivative (PD) controller that drives the DC motor using Pulse Frequency Modulation (PFM) to accomplish the control objective. The results show an accurate, real-time and efficient controller design.**

I. Introduction

The inverted pendulum problem is usually taught during automation control courses. It has most of the features needed to learn control techniques: a dynamic plant to control the pendulum, sensors to provide feedback, and actuators to correct or stabilize the pendulum position. This paper presents a controller for a robotic platform similar to the inverted pendulum scenario. In our case, a pencil is placed on a moving cart driven by two motors (one per each axis) [1].

To control a robotic platform, actuators and sensory devices are needed. Classic approaches use a Proportional-Integral-Derivative (PID) controller, where the error signal is the input to the controller. In our setup, the error signal is the angle of the pencil from the vertical position. The novelty of this paper is the way the feedback is computed and how the information is processed to apply the control. There are some similar previous works: in [2], the event-based PID controller used in this work was presented. They used the encoders of the motors to provide the feedback to the controller. In [3] a full event-based architecture was presented. Nevertheless, it was an open-loop technique. We propose to use two kinds of sensors to provide the feedback: Address-Event-Representation (AER) retinas [4]–[6] as the event-based sensors and potentiometers as classical sensors. Besides, the controller is implemented on a hardware platform (FPGA) to achieve a minimum delay as it is the main requirement for a pendulum to be controlled.

Previous work has demonstrated that a pencil can be balanced with vision sensors. This task was first accomplished using dynamic vision sensors and jaer [7], [8]; i.e. the computation was done with software using a USB interface to the DVS cameras. Next it was implemented in fixed point code running on the NXP 32-bit microcontrollers on the current pencil balancer robotic platform [1]. In this paper, the challenge is to demonstrate that these same computations can be implemented in logic circuits implemented on an FPGA, by developing a DVS sensor post-proceser and a motor controller on the FPGA. This way, no computer will be needed and events can be processed with the absolute minimum system latency.

The paper is structured as follows: first the methodology used is shown. Next, the dynamics of the plant (pencil balancer robotic platform) are detailed. Follows a section where simulation results of an ideal PD controller are shown. Then, a description of the setup robotic platform and the controller to implement on the FPGA is presented. Next, the results are shown. Finally, the discusion and conclusion sections are placed.

II. Methodology

We follow the classic methodology of automation control. That is, first, a mathematical model of the pencil balancer dynamics is created. Once the model is ready, we use Simulink by Mathworks to simulate and validate our model and eventualy our proposed controller. Then, the controller is simulated and its parameters are tuned, double checked by theoretical approaches (i.e. all the poles of the close-loop transfer function are in the left half plane), and the behavior of the entire system is checked. Finally, the system is implemented on the FPGA, and the performance is measured using the data from the hardware.

III. System dynamics

These are the equations of the dynamics of the pencil [9] where $x(t)$ is the position of the cart, $u(t)$ is the external force we can apply to the cart and $\theta(t)$ is the angle of the pencil from the vertical:

$$(M+m)\frac{d^2}{dt^2}x(t)+\gamma\frac{d}{dt}x(t)=$$
$$u(t)+ml\sin\theta(t)(\frac{d}{dt}\theta(t))^2-ml\cos\theta(t)\frac{d^2}{dt^2}\theta(t) \quad (1)$$

$$(I+ml^2)\frac{d^2}{dt^2}\theta(t)=mgl\sin\theta(t)-ml\cos\theta(t)\frac{d^2}{dt^2}x(t) \quad (2)$$

The term γ included at the first equation models the friction of the cart. The I represents the angular momentum of the pencil. M and m are the mass of the cart and the mass of the pencil respectively. l is the length of the pencil and g is gravity.

A. Linearized Equations

It is reasonable to think that the angle will stay close to zero, therefore the equations can be linearized by simplifying them with the following hypothesis:

- $I = 0$ (center of mass is equal to the center of gravity)
- $\sin\theta(t) = \theta(t)$
- $\cos\theta(t) = 1$
- $(\frac{d}{dt}\theta(t))^2 \approx 0$

Then, the resulting equations are:

$$(M+m)\frac{d^2}{dt^2}x(t) + \gamma\frac{d}{dt}x(t) = u(t) - ml\frac{d^2}{dt^2}\theta(t) \quad (3)$$

$$l\frac{d^2}{dt^2}\theta(t) = g\theta(t) - \frac{d^2}{dt^2}x(t) \quad (4)$$

B. Space state equations

In order to simulate the equations with different initial conditions, we operate with the linerized equations to get them in the form of the space state equations:

$$\dot{x}(t) = A \times x(t) + B \times u(t) \quad (5)$$

$$y(t) = C \times x(t) + D \times u(t) \quad (6)$$

From Eq. (4):

$$\frac{d^2}{dt^2}x(t) = g\theta(t) - l\frac{d^2}{dt^2}\theta(t) \quad (7)$$

Then, using Eq. (7) we can substitute into Eq. (3):

$$\frac{d^2}{dt^2}\theta(t) = \frac{-1}{lM} \times (u(t) - (M+m)g\theta(t) - \gamma\frac{d}{dt}x(t)) \quad (8)$$

And going back to (7):

$$\frac{d^2}{dt^2}x(t) = \frac{1}{M}u(t) - \frac{\gamma}{M}\frac{d}{dt}x(t) - \frac{mg}{M}\theta(t) \quad (9)$$

With the expressions (8) and (9), we have obtained the state space definition for our system:

$$\frac{d}{dt}\begin{bmatrix}\theta(t)\\ \frac{d\theta(t)}{dt}\\ x(t)\\ \frac{dx(t)}{dt}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ \frac{g(M+m)}{lM} & 0 & 0 & \frac{\gamma}{lM}\\ 0 & 0 & 0 & 1\\ \frac{-mg}{M} & 0 & 0 & \frac{-\gamma}{M}\end{bmatrix} \times \begin{bmatrix}\theta(t)\\ \frac{d\theta(t)}{dt}\\ x(t)\\ \frac{dx(t)}{dt}\end{bmatrix} + \begin{bmatrix}0\\ \frac{-1}{lM}\\ 0\\ \frac{1}{M}\end{bmatrix} \times u(t) \quad (10)$$

$$\begin{bmatrix}\theta(t)\\ x(t)\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix} \times \begin{bmatrix}\theta(t)\\ \frac{d\theta(t)}{dt}\\ x(t)\\ \frac{dx(t)}{dt}\end{bmatrix} \quad (11)$$

The calculated matrices are:

$$A = \begin{bmatrix}0 & 1 & 0 & 0\\ \frac{g(M+m)}{lM} & 0 & 0 & \frac{\gamma}{lM}\\ 0 & 0 & 0 & 1\\ \frac{-mg}{M} & 0 & 0 & \frac{-\gamma}{M}\end{bmatrix} \quad B = \begin{bmatrix}0\\ \frac{-1}{lM}\\ 0\\ \frac{1}{M}\end{bmatrix} \quad (12)$$

$$C = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix} \quad D = 0 \quad (13)$$

IV. Simulations

To simulate these equations, we have used Simulink by Mathworks to simulate the system shown in Fig. 1.

Firstly, we simulate the model described including a classic Proportional-Derivative (PD) controller to check the behaviour of the pendulum with respect to its cart. If the results are good enough (the position of the cart can be controlled or at least estimated), the controller can then be simulated considering the restrictions on the implemented version of the controller on the FPGA.

A PD controller includes two tunable parameters: a proportional gain K_p that multiplies the error signal, and a derivative gain K_d that multiplies the derivative of the error signal with respect to time.

The initial conditions are randomly generated for both cart position and angle tilt to assure a correct performance regardless of the starting point.

Fig. 2 shows the first step taken to stabilize the pencil. We have first fixed a high value for K_p (higher than 40) to provoke a twitch of the pencil to let the motors act. The K_d is fixed to half of K_d. After some trials, we achieved a stabilized vertical pencil with a minimum movement of the position of the cart (values for the parameters: $K_p = 48$ and $K_d = 24$). This small cart-motion result leads us think that it could be possible to control the angle despite the cart position.

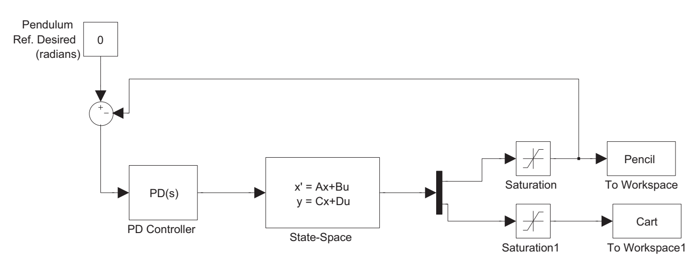

Fig. 1. Simulink setup to simulate the State Space equations. The plant, controller, and reference are shown. The saturation blocks at the output of the controller are placed to avoid exceeding the limits of the robotic setup.

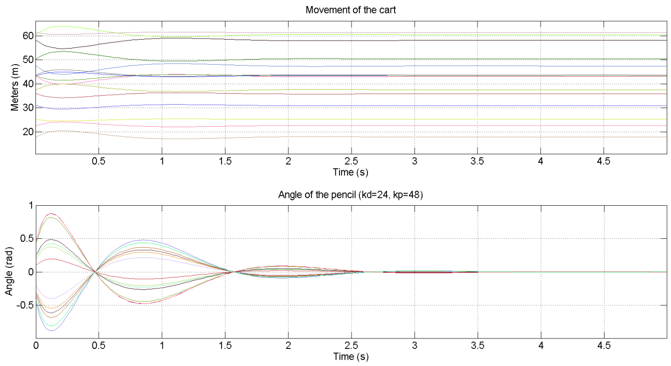

Fig. 2. Simulation results for the cart position (upper) and angle of the pencil (lower). The PD controller is tuned with the values: $K_p = 48$ and $K_d = 24$. The upper plot represents the movement of the cart and the lower plot the angle of the pencil in radians. We have randomly set twenty different initital conditions (cart position and angle from the vertical) to check the behaviour of the platform.

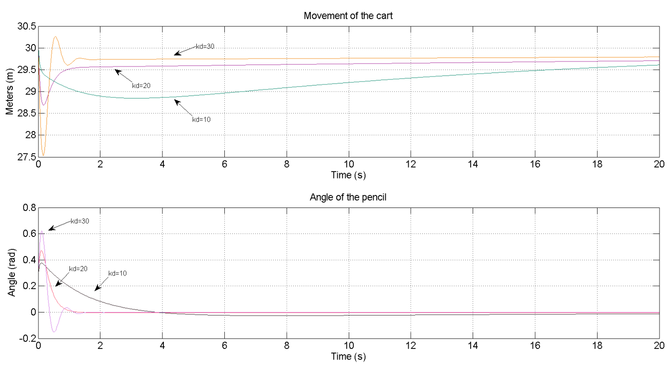

Fig. 3. Simulation results for the cart position and angle of the pencil when the K_p is fixed to a value of 18 and the K_d is changed within the set {10,20,30}. It can be observe that, the higher the value of K_d the more overdumped the response becomes.

Then, to avoid an underdamped or a critically damped response and to reduce the initial twitch, we reduced the value of K_p. Fig. 3 shows how, with a $K_p = 18$, if the value of K_d increases, the response is faster but becomes critically damped when the parameter is higher than $28 - 29$. The minimun K_d parameter found is 6 (the controller takes nearly 60 seconds to stabilize the system); a lower value will provoke the pencil to fall.

If one looks at Figs. 2 and 3, the movement of the cart is minimun with respect to the initial random value. This data confirms that the control can be done taking into consideration only the angle of the pencil. Therefore, the initial position the cart should be as closer as possible to the final position desired.

The controller already presented is the ideal one. However, it is not easy to measure the angle of the pencil within the robotic platform. Therefore, to make it clear, we first present the robotic platform and then show the control will be done on the FPGA.

V. SETUP

The 'Pencil Balancer' robotic platform is shown in Fig. 4. It is comprised of both sensors and actuators:

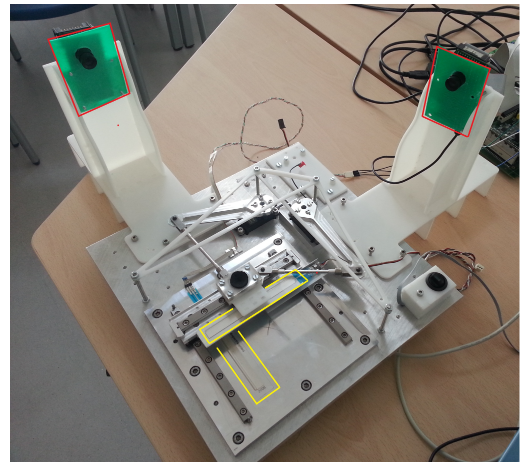

Fig. 4. First version of the pencil balancer robotic platform. Both white cases are used to fix the retinas; the green piece is used to adapt the white case to that particular model. It can be seen the cart, the potentiometers sensors (yellow squared), the arms actuated by the motors and the old microcontroller with the controller [1].

- AER Retina sensors [10].
- Two potentiometer strips placed at the table base. These sensors can supply information about the position of the cart. The reference of the sensor is: SpectraSymbol SoftPot SP-L-0100-101-ST.
- The actuators are both DC motors (servomotors modified to become DC motors). The motors used are from the company Futaba.

As seen in Fig. 4, the robotic platform has two axes. Since both axes have the same configuration: one retina, potentiometer and motor, we duplicate the PD controller.

VI. CONTROLLER TO IMPLEMENT ON THE FPGA

To implement the controller on the FPGA, it is needed to measure the angle of the pencil. To do that, the proposed controller to implement on the FPGA is based on two different types of sensors: AER retinas and soft potentiometer sensor bars that supply the cart position. We use both sensors to provide the appropiate feedback to the controller, i.e. the angle:

- AER Retina: The AER retina will provide the visual information (pixel activity) to a VHDL component synthesized on the FPGA. It will firstly filter background activity not spatially or temporally correlated, and secondly, it will track the center of mass of the object in the plane of view. Many parameters can be configured to fine tune the filtering and tracking processes [11].
- The potentiometer is a resistive sensor that provides an analog value depending on where the cart is placed at each moment. This value is fed into an ADC and eventually to the FPGA using an Serial Peripheral Interface (SPI) protocol.

The FPGA system substracts these two outputs to get the sine of the angle. Given our approximation $sin(\theta) \approx \theta$, we can use the result of that subtraction divided by the length of the

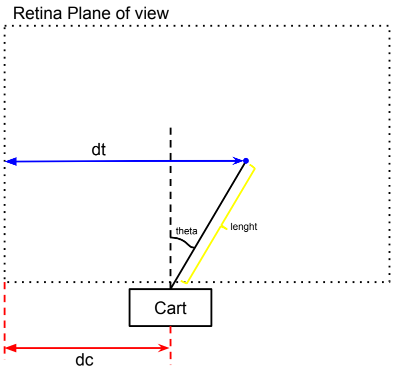

Fig. 5. Illustration of angle measurement with the FPGA. The potentiometer provides the cart position: d_c and the upper tracker of the vision algorithm supplies the pencil distance: d_t. The range of both parameters goes between $0 - 127$.

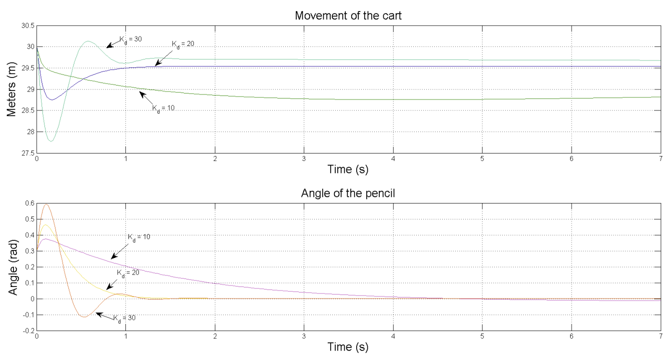

Fig. 6. Simulation results for the cart position (upper) and angle of the pencil (lower) when the simulation is done under the design requirements of the FPGA.

pencil as the error signal, or angle measured, to supply to the controller (Fig. 5):

$$sin(\theta) \approx \theta = \frac{d_t - d_c}{l} \tag{14}$$

Then, the angle can be supplied to the event-based PD controller implemented on the FPGA [2]. Since the platform is divided into two axes, we have replicated the controller. Thus, we will have a retina, a potentiometer and a motor for each axis.

To validate the proposed controller, simulations are carried out with the Simulink setup modifying how the feedback is supplied, i.e. how the value of the angle is calculated. In this case, the results are slightly different for the previous one. Fig. 6 shows the result using the same parameters of Fig. 3 and it looks exactly the same but a litle underdamped with respect to the previous experiment. The results show also a minimum movement of the cart from its initial position.

These results validate our proposal for computing the feedback signal to be supplied to the controller.

VII. RESULTS

One of the main parts of the system is the controller. It is implemented using the AER Node board [12]. This board includes a Xilinx Spartan-6 LXT 1500 FPGA. It was developed by RTC lab and IMSE under the VULCANO project[1] and it allows high speed serial AER communications over Rocket IO transceivers, and adaptation to particular scenarios using daughter boards connected on top. The setup includes a daughter board with a USB microcontroller (Reference C8051f320 from Silabs) that communicates with the FPGA using SPI. This interface is used to send the parameters needed for each block of the controller [2] and to send the converted values from the potentiometer. To convert the analog values read from the potentiometer, a 10-bit ADC included in the microcontroller is used. The values measured are truncated to obtain a range of (0-127).

Fig. 7 shows a snapshot taken when the tracker is running. The information about the center of mass provided by the tracker is sent to the controller within a configurable time depending on the events belonging to the initial cluster fixed [11]; however, it usually provides a new center of mass event every $4ms$. Fig. 8 shows a capture of the chipscope (tool from Xilinx to monitor the real signal in the FPGA) where the values from the potentiometer, from the tracker (once averaged) and the subtraction of both can be monitored.

The event-based PD controller produces events that last 20 ns (the period of the clk of the AER Node board). Since we are using rate-coded information, the modulation used to drive the motors is Pulse Frequency Modulation (PFM). This modulation uses a square-signal with a fixed duty-cycle and a message encoded by frequency. Therefore, the outcoming events from the controller can be used to drive the motors after they are time-extended. Otherwise, the events can be filtered by the motors.

The potentiometer has a length of 10 cm. It provides a step resolution of 10 cm $/128$ steps $= 0,078125$ cm per step. This is the maximum resolution on the cart position.

Our FPGA system updates the position of the cart with a maximum delay of 24μs. The ADC of the microcontroller converts the potentiometer reading in 2μs, so the SPI transmission of the value takes 22μs to be completed.

VIII. DISCUSION

Previous work [1] claimed a maximum delay of $10ms$. With our technique to provide the feedback we improve the time up to $4ms$ as maximum. So, the feedback loop is very fast. The main problem of this system could be that the DC motors do not have encoders to ensure that the commanded position is reached. Although we consider that they are not needed because we are not controlling the position, the addition of the encoders is a reasonable step to take to continue with this project (to make a comparison with the potentiometer value and assure that the commanded position is reached). Conversely, the motors must have a very fast response to follow the events provided to them. In any case, by using PFM we do not have to consider the PWM time (usually a delay of 20 ms).

[1] VULCANO Project: Ultra-Fast Frame-less Vision by Events. Application to Automation and Anthropomorphic Cognitive Robotics. (TEC 2009-10639-C04-02)

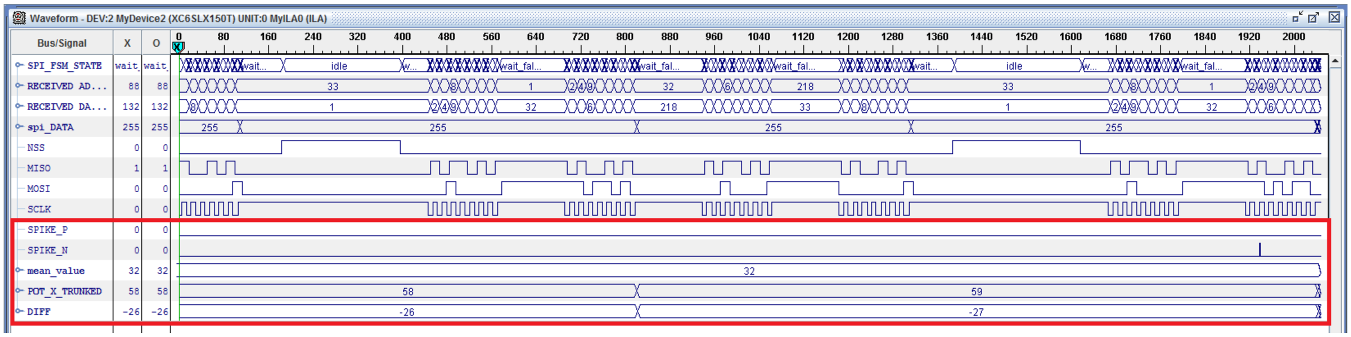

Fig. 8. Capture taken from the chipscope tool provided within the Xilinx ISE Design Suite. The upper signals monitored are the SPI communication channel. The last five provide useful information: spike_p and spike_n are the output events from the controller. These lines will be used to drive the motors after lengthening them. The positive or negative event will depend on the sign of the sin value computed. The last three rows represent the sin computation: first, the average value from the tracker is shown, then the truncated value of the potentiometer is shown, and finally the subtraction between both (DIFF signal).

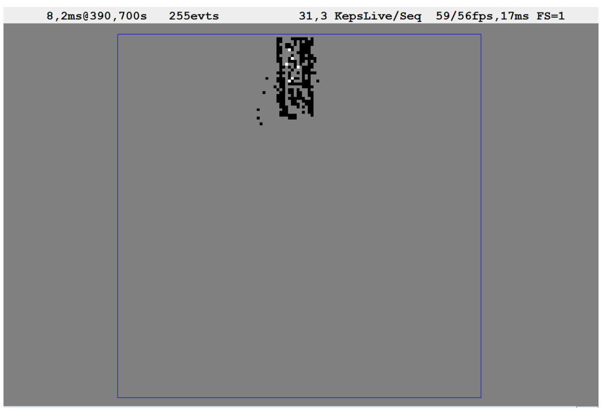

Fig. 7. The snapshot is taken from the jAER [8] viewer. The enabled tracker is using a window of 128 x 32 pixels, i.e. the first horizontal space if we divide the retina view in four parts. The black active pixels show a capture of the pencil while it is moving. The white pixels represent the center of mass of the pencil computed at that time.

IX. Conclusion

We have presented a design of an event-based controller system including sensory feedback from neuromorphic sensors and from classical sensors to control the pencil balancer robotic platform. The presented system on the FPGA has a maximum feedback update time of 24μs. Every 24μs, a new update at the cart position will be ready. The 10 bits ADC of the microcontroller needs $2\mu s$ to convert the potentiometer analog data to a digital value before starting the transmission to the FPGA. On the side of the AER retinas, the center of mass is updated every $4ms$. We have improved previous works up to 60% in terms of real-time computation.

References

[1] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. Douglas, and T. Delbruck, “A pencil balancing robot using a pair of aer dynamic vision sensors,” in *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, May 2009, pp. 781–784.

[2] A. Jimenez-Fernandez, G. Jimenez-Moreno, A. Linares-Barranco, M. J. Dominguez-Morales, R. Paz-Vicente, and A. Civit-Balcells, “A neuro-inspired spike-based pid motor controller for multi-motor robots with low cost fpgas,” *Sensors*, vol. 12, no. 4, pp. 3831–3856, 2012.

[3] F. Perez-Peña, A. Morgado-Estevez, A. Linares-Barranco, A. Jimenez-Fernandez, F. Gomez-Rodriguez, G. Jimenez-Moreno, and J. Lopez-Coronado, “Neuro-inspired spike-based motion: from dynamic vision sensor to robot motor open-loop control through spike-vite,” *Sensors*, vol. 13, no. 11, pp. 15 805–15 832, 2013.

[4] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, “A 240 x 180 130 db 3 us latency global shutter spatiotemporal vision sensor,” *Solid-State Circuits, IEEE Journal of*, vol. 49, no. 10, pp. 2333–2341, Oct 2014.

[5] T. Serrano-Gotarredona and B. Linares-Barranco, “A 128x128 1.5sensitivity 0.9sensor using transimpedance preamplifiers,” *Solid-State Circuits, IEEE Journal of*, vol. 48, no. 3, pp. 827–838, 2013.

[6] P. Lichtsteiner, C. Posch, and T. Delbruck, “A 128x128 120 db 15 us latency asynchronous temporal contrast vision sensor,” *Solid-State Circuits, IEEE Journal of*, vol. 43, no. 2, pp. 566–576, 2008.

[7] J. Conradt, P. Lichtsteiner, R. Berner, T. Delbruck, R. Douglas, and M. Cook, “High speed pole balancing with only spike-based visual input,” in *Neural Information Processing Systems Conference (NIPS)*, 2008.

[8] T. Delbruck. Java tools for address-event representation (AER) neuromorphic processing (jAER) open source project. Http://jaer.wiki.sourceforge.net/.

[9] K. Ogata and Y. Yang, *Modern control engineering*, 5th ed. Prentice Hall, September 2009.

[10] C. Posch, T. Serrano-Gotarredona, B. Linares-Barranco, and T. Delbruck, “Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output,” *Proceedings of the IEEE*, vol. 102, no. 10, pp. 1470–1484, Oct 2014.

[11] F. Gomez-Rodriguez, L. Miro-Amarante, F. Diaz-del Rio, A. Linares-Barranco, and G. Jimenez, “Real time multiple objects tracking based on a bio-inspired processing cascade architecture,” in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, May 2010, pp. 1399–1402.

[12] T. Iakymchuk, A. Rosado, T. Serrano-Gotarredona, B. Linares-Barranco, A. Jimenez-Fernandez, A. Linares-Barranco, and G. Jimenez-Moreno, “An aer handshake-less modular infrastructure pcb with x8 2.5gbps lvds serial links,” in *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on*, June 2014, pp. 1556–1559.