Performance Analysis of a Wireless Camera Network Node

Leonardo Gasparini
Dept. of Information Engineering and Computer Science
University of Trento
38123 Povo (TN), Italy
gasparini@disi.unitn.it

Roberto Manduchi
Dept. of Computer Engineering
University of California
Santa Cruz, CA 95064, USA
manduchi@soe.ucsc.edu

Massimo Gottardi
Smart Optical Sensors and Interfaces
Fondazione Bruno Kessler
38123 Povo (TN), Italy
gottardi@fbk.eu

Dario Petri
Dept. of Information Engineering and Computer Science
University of Trento
38123 Povo (TN), Italy
petri@disi.unitn.it

***Abstract*—The lifetime of the nodes of a wireless camera network is often limited to some hours or some days. In this work we analyze the performance of a custom wireless node equipped with an ultra low power imager. We prove that the employment of non-standard video sensors is a key point in the expansion of camera-based WSNs.**

I. Introduction

The use of wireless sensor networks for video surveillance applications has received considerable attention by the research community. Power consumption is arguably one of the most critical characteristics of a wireless node. For example, consider the following scenario. A network of miniaturized wireless camera nodes is deployed to monitor a wide, remote area. The network needs to be operational without human intervention for 30 days. Assume that each node operates on a pair of batteries that approximately supply $2200\,\mathrm{mAh}$ at $3\,\mathrm{V}$, the lifetime requirement implies that each node should draw no more than 3 mA on average for both image acquisition and processing, and possibly transmission of relevant data. This is a very tight constraint. To our knowledge, no system reported in the literature can operate with such little power: for example, Cyclops [8], MeshEye [6], and Citric [7] nodes exceed the constraint by two orders of magnitude.

Imaging sensors are typically designed "for humans", providing possibly high resolution images with several bits per pixel. An imager can be thought of as a multi-dimensional array of sensors. Thus, it may be much more demanding in terms of energy per acquisition with respect to other simpler devices like temperature and pressure sensors. More importantly, it generates a large amount of data which needs to be processed with stringent time constraints, resulting in high power requirements.

In fact, typical surveillance algorithms do not need the whole image being acquired: in many cases, only particular image features (such as edges, corners and brightness or color histograms) are of interest. As the remainder of the information is ignored, the energy required to gather it is effectively wasted.

In recent years, a different approach to the design of video sensors have been proposed by the researchers [2], [9], one that attempts to limit power requirements by implementing the processing (or some parts of it) directly *on chip*. The idea is to design not mere light sensors, but devices which sense a specific visual property of the scene. In such a way, less data is generated by the chip, resulting in a reduction of the signal activity at the output pins, and, most importantly, the algorithm can be executed with no or little pre-processing.

We are developing a new module for video surveillance applications that can function with extremely small power requirements (around $3\,\mathrm{mW}$). This module is comprised of an ultra-low-power sensor and a flash-based Field Programmable Gate Array (FPGA). The FPGA manages the system and processes the data generated by the sensor, which provides binary images representing the points with high brightness contrast. Although communication with other devices is currently performed by a serial interface, we are planning to add a ZigBee module for wireless communication.

In order to demonstrate the capabilities of this module, we have configured it for application as a people counter. In contrast with traditional systems based on optical traps, which require the installation and calibration of a light beam and an optical receiver, a camera-based system for people counting can be easily and quickly deployed in any environment, representing and ideal mechanism for impromptu installation. In this paper we analyze the performance of the system and estimate the power consumption and lifetime of the node.

The paper is organized as follows. In section II we describe the components that are part of the current prototype. In section III we discuss the implemented algorithm. The power consumption analysis is then carried out in section IV.

II. Node architecture

The node is mainly composed by three elements: the imager, the FPGA and a transceiver, as shown in figure 1.

A. Video-sensor

The imager employed in our system is a prototype developed at the Fondazione Bruno Kessler of Trento (Italy) [1]. This is a 128×64 pixels, binary, contrast-based sensor which provides address-based output data asynchronously. The imaging operation is obtained by a two-stage process: acquisition of the current frame, and generation of the output frame. During

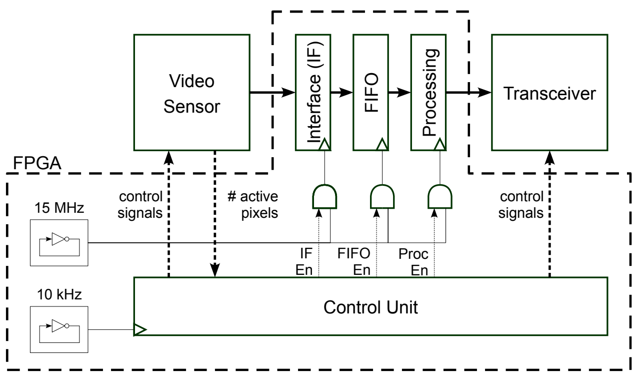

Fig. 1. Block scheme of the node.

the acquisition process, each pixel receives a binary value obtained by comparison of the incoming irradiance at three locations: the pixel itself, the pixel above and the pixel at its right. We refer to this L-shaped structure as the *kernel*. The pixel is said to be *"active"* if the difference in incoming light between the most and the least irradiated pixels in the kernel exceeds a predefined amount. Thus, at the end of the acquisition process, the active pixels represent the parts of the image characterized by high contrast.

In the second part of the process, the sensor computes the pixel-by-pixel (bit-by-bit) difference of the current image with a previously acquired (binary) frame stored in the internal memory. The sensor then waits for an external command to start the read-out process, in which the column address and the sign of the non-zero pixels of the difference image are provided at the output pins (the row address is derived from an End-Of-Row signal). This process is asynchronous, meaning that this "difference image" is raster-scanned, and an output enabling signal running at $80\mathrm{MHz}$ is raised every time the data at the chip's output pins represents the address of a non-zero pixel. This process is executed in less than $200\,\mu\mathrm{s}$.

A number of characteristics contribute to the power efficiency and performance of this system. By subtracting a previously acquired image from the current frame it is possible to implement background subtraction and basic motion detection on-chip, rather than on an external processor. The mechanism of address-based image coding in most cases reduces the amount of data to be transferred outside the chip. Moreover, the frame rate can be quite high, since it is only limited by the duration of the acquisition process. Under normal light conditions the integration time can be as low as $5\,\mathrm{ms}$, enabling a frame rate as high as $200\,\mathrm{frames/s}$.

In addition to this mode of functioning, named *Active*, in order to reduce power consumption even further, the sensor supports a low-power mode, called *Idle mode*. When this mode is activated, the chip provides at the output pins only the number of non-zero pixels present in the difference image, rather than the whole binary image. This is particularly useful when long periods of low activity are expected, a typical situation in surveillance applications. When nothing changes in the scene, the difference image is for the most part equal to zero. Thus, the sensor may be set to Idle mode (thus drawing much less power than in Active mode) until the number of non-zero pixels in the difference image is above a certain threshold, at which point the sensor is set to Active mode to enable detailed image analysis.

The overall power consumption of the sensor is extremely small. At a frame rate of $50\,\mathrm{fps}$, with 25% active pixels, the sensor draws approximately $100\,\mu\mathrm{W}$ when in Active mode. Note that since active pixels represent high contrast image portions (typically edges), it is unlikely that more than 25% of the pixels are active. If the system is set to Idle mode, the power consumption reduces to $30\,\mu\mathrm{W}$. This is over two orders of magnitude less with respect to the image sensors available on the market.

B. Storage and Processing Unit

Our module uses a FPGA-based board for control of and communication with the imager and the transceiver. FPGAs have well known advantages in terms of speed and processing power with respect to micro-controllers, which represent the typical solution for the management of standard wireless network nodes. On the other side, FPGA dissipate much less power than high performance embedded processors such as the Intel XScale family, which has been employed in several camera-based motes [3], [7]. For our implementation we selected Actel IGLOO M1-AGL600, which is characterized by $600\,\mathrm{k}$ system gates. IGLOO is a family of flash-based FPGA featuring two important characteristics for our application: static power consumption on the order of tens of $\mu\mathrm{W}$; and the capability to be live at power up [5].

Within the FPGA we have implemented a ring oscillator, a Firs Input First Output (FIFO) memory with its interface, a processing unit and a controller. Two clock domains are implemented: one running at a low frequency ($\sim 10\mathrm{kHz}$), and one running at a higher frequency ($\sim 15\ \mathrm{MHz}$), both derived from the ring oscillator's output which is divided through a clock divider chain. The first clock (which runs continuously) provides the time base used for the control of the whole node.The second clock is needed to receive the acquired image from the camera. It can also be used to clock the processing unit, for applications with particularly demanding processing requirements. We will refer to this element as the *Acquisition and Processing Clock (A/P Clock)*.Significant power saving is achieved by disabling this clock when not needed. This task is performed by the control unit through an AND gate. Since the components implemented in the FPGA core need to be enabled in different periods of time, a dedicated gate is provided for each controlled component.

The control unit generates the timing signals for the camera, which then provides either the image or the number of active pixels at its output pins. In the first case, the data represents the column address of the active pixels and flows at a maximum rate of $80\ \mathrm{MHz}$ in bursts of no more than 128 elements (i.e. the image is transferred one row at a time). Unfortunately, such a high speed can not be supported by big soft memories

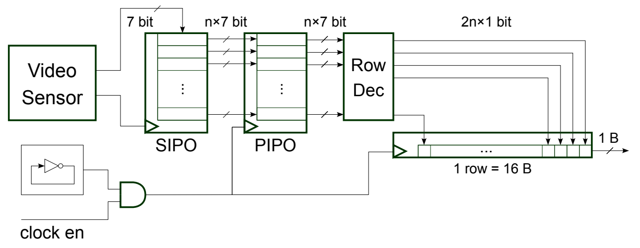

Fig. 2. Block scheme of the data interface.

implemented in the FPGA, while adding an external FIFO would increase the node's complexity and power consumption.

In order to cope with the transmission speed and slow down the access to the memory, the data goes through an interface implementing the cascade of a serial to parallel converter and a decoder, as shown in Fig. 2. This is obtained by means of a Serial Input Parallel Output (SIPO) memory, with a number of elements n high enough that the corresponding synthesized circuit can run at the speed of the sensor: the higher n, the lower the minimum frequency supported. Then, the data is copied into a Parallel Input Parallel Output (PIPO) memory, clocked at a lower frequency, in order to allow the SIPO memory to be overwritten without data loss. The addresses in the PIPO memory undergo the decoding process, through which we reconstruct the current row of the acquired image, which is stored into another buffer. This data is then safely transferred into the FIFO memory when the End-Of-Row signal is asserted. This mechanism was shown to be reliable as long as the A/P Clock's frequency is greater than $80\,\mathrm{MHz}/n$. The soft FIFO is an 8Kbit memory, in which the whole acquired image is stored. The data is currently packed in bytes, although the system can be configured for different word sizes. Once the read-out process has been completed, the unit processes the image to generate the desired output, which is then sent to the transceiver for wireless transmission.

When the camera is set in Idle mode, the data at its output pins is simply a string of bits representing the number of active pixels. As there is no high-speed data flow nor intensive processing to handle, the A/P Clock is never needed in Idle mode.

In our prototype we have implemented the following policy regarding the management of the Active and Idle modes of the camera:

- As long as the number of active pixels remains below a threshold n_p^{th} (which depends on the scene being monitored), the camera is set in Idle mode;
- When the number of active pixels exceeds n_p^{th}, meaning that something relevant is happening in the scene, the sensor is tasked to provide the entire image to the processing unit by setting its output mode to Active;
- The camera is switched back to Idle mode only when the number of active pixels stays below n_p^{th} continuously for a period t_{idle}, whose duration is application-dependent.

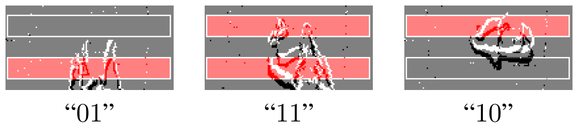

Fig. 3. An image sequence acquired by the sensor with two VILs drawn and the associated system state. The sequence represents a person walking upwards. Each VIL activates when the number of non-zero pixels within it exceeds a given threshold. The system state is represented by a two-bit variable, describing the status of the VILs.

C. Transceiver

Data is currently transmitted to a host computer through a RS-232 interface, but we are planning to install a wireless module such as ChipCon CC2420 IEEE 802.15.4-compliant $250\,\mathrm{kbps}$ radio. It draws a current of less than $20\,\mathrm{mA}$ when transmitting or receiving. Conversely, consumption reduces to less than $0.5\mathrm{mA}$ when in idle mode and around $20\mu\mathrm{A}$ in power down mode [12]. Thus, power efficiency is achieved only if the control unit exploits massively the low power modes of the wireless module. This implies that no image is sent over the network, but only the output of the implemented algorithm, and that communication occurs at a very low rate.

III. People Counting Algorithm

In order to demonstrate the capabilities of this system in a realistic scenario, we configured it to function as a "people counter". The camera is placed on the ceiling of a passageway, facing downwards, and registers the passage of people in both directions. The algorithm exploits the ability of the sensor to detect motion by comparing each acquired binary image with the previous frame, stored in the imager's internal memory. Currently the algorithm is designed so as to detect the passage of individual persons, but we are working on it to be able to discriminate a wider range of situations.

The difference images provides information about high contrast points undergoing motion: positive and negative active pixels of the difference image represent moving edge points in the current or previous frame respectively.

When the sensor detects no activity for a predefined amount time t_{idle}, the controller is programmed to enable the Idle mode of the sensor and disable the A/P Clock to save energy. As soon as an event is registered in the scene (as measured by the number of active pixels), the controller activates the sensor and the other elements to process the whole image.

Detection of persons transiting in a direction or the other is performed by assigning a "state" to the entire system at each frame and recording state transitions through time. Each state gives an extremely concise representation of the presence and location of a person in the camera's field of view. We use the idea of *Virtual Inductive Loops (VILs)* [10], which are non-overlapping windows defined in the image. If enough active pixels are found within a VIL, the VIL is said to be *"Active"*, otherwise it is *"Off"*. At a given instant (frame), the state of the system is represented by the values of the VILs defined in the image (as shown in Fig. 3).

TABLE I
CLASSIFICATION PERFORMANCE

training set	video #1	video #2
training events	154	153
test set	video #2	video #1
test events	153	154
correct detections	151	151
errors	0	1
missed detections	2	2
false alarms	0	19

The state evolution is modeled as a Markov Process of the second order. The transition probabilities for each "event" (such as a person transiting in a certain direction) are learnt during a training phase with manually labeled sequences. At run time, sequences ("segments") of states are identified, and then are grouped into "intervals" corresponding to different events according to a Maximum Likelihood criterion. This is accomplished by a modified version of the Dijkstra's algorithm, with a complexity of $O(n_{st})$, where n_{st} represents the number of state transitions that occurred since the last time the sensor has been activated. The linear dependence on n_{st} is caused by the fact that the algorithm looks for the most likely combination of events among all the possible combinations: intuitively, at every new state transition the number of possible combinations that we have to take into account doubles, since the new segment may represent a new event, or may be part of the previous one.

Our current implementation uses two VILs, and we represent probability values as 8 bit variables. We found that the linear relationship between the number of clock cycles required to execute the whole algorithm and n_{st} depends on a constant $c = 126$. By incrementing the number of VILs and the number of bits for data representation, we can achieve better performance at the expense of a higher complexity.

Note that the algorithm does not need to be executed at every frame, but only at every state transition, meaning that there is no need to activate the processing unit when the system state has not changed.

A. Classification performance

We measured the classification performance of our algorithm by recording two long videos and alternating them in the role of training set and test set. The results are shown in Tab. I. In one case, almost 99% of the detections were correct, while in the other the amount of false alarms is very high. This is due to the presence of long shadows on the ground which the sensor is unable to filter out. Since the shadows are present in the test set but not in the training set, the system sees a person and his/her shadow as two persons walking in close sequence.

IV. NODE LIFETIME ESTIMATION

We measured the power consumed by the system, with the FPGA mounted on a commercial development board equipped with several components in addition to the M1-AGL600: a 1 MB SRAM, a 16 MB Flash memory, a crystal oscillator, a programmer, a USB to RS-232 converter chip, expansion connectors, leds and switches. We connected the camera to the board through one of the available connectors. The node communicates to a host computer through a USB connection that implements the RS-232 interface. The board allows for measurement of the current flowing through the FPGA core, its I/O banks and the camera separately.

Measurements were carried out by forcing the system to run in one of the following configurations:

1) Idle mode, entered when no motion is detected for a period longer than t_{idle};
2) Active mode, with image acquisition but without executing the algorithm, enabled when something relevant is taking place (active pixels detected in a VIL), but no system state transition occurs;
3) Active mode, with image acquisition and execution of the algorithm, activated when the system state changes.

The value of t_{idle}, i.e. the amount of time of low pixel activity before the node enters the Idle mode, is chosen so as not to allow the system to be switched in Idle mode while an event is still happening, thus avoiding unwanted breaks in the Markov chain.

When the camera is in Idle mode, only the number of active pixels is provided at the sensor's output pins, and therefore there is no need to enable the clock for the interface, the FIFO memory or the processing unit, i.e. the AND gate depicted in Fig. 1 is closed. The node enters the Active mode as soon as the number of active pixels in the acquired image is higher than a chosen threshold. When this condition occurs, the system state needs to be continuously monitored. This is achieved by enabling the clock of both the interface and the FIFO memory, so storing the image; once the image has been stored, the interface clock is disabled. Since the state evaluation is performed by the processing unit, also the related clock needs to be enabled. The execution of this task is almost immediate and if no transition occurs, the algorithm exits and both the FIFO and the processing units' clocks are disabled until the next frame is acquired; conversely, the processing unit is kept alive until the algorithm reaches its end.

On the basis of these considerations, we analyzed both power and timing performance of our system. The parameters that describe the algorithm from a timing point of view are: the frame rate r_{frame}, the complexity-related constant c (defined in section III), and the frequency $f_{AP\ Clk}$ of the clocking signal provided as input to the processing unit. From this parameters we can extract the average duty cycle D of the processing unit within a frame, and the maximum number of state transitions $n_{st,max}$, allowed by the FPGA before it fails (meaning that a new frame arrives before the previous frame has been fully analyzed). Given the time per frame $t_{frame} = 1/r_{frame}$ and the time needed for executing the algorithm $t_{proc}(n_{st}) = n_{st} \cdot c / f_{AP\ Clk}$, the average duty cycle is computed as:

$$D = \frac{\bar{t}_{proc}}{t_{frame}} = \frac{\bar{n}_{st} \cdot c \cdot r_{frame}}{f_{AP\ Clk}}$$

where $\bar{n}_{st}$ represents the average number of state transitions. Its value depends on the number of VILs being used and on the distribution over time of the flow of people across the monitored area. D relates the average power consumed by the system operating in Active mode when executing the algorithm ($\bar{P}_{A,proc}$) to the case when it just computes the current state ($\bar{P}_{A,acq}$) as follows:

$$\bar{P}_{A,proc} = \bar{P}_{A,acq} + k_{proc} \cdot D$$

proc where k_{proc} is a constant dependent on the hardware configuration. We can consider $\bar{P}_{A,acq}$ as the amount of power required to acquire the image and store it inside the FPGA, while $k_{proc} \cdot D$ represents the average power required for the execution of the algorithm.

The second parameter, $n_{st,max}$, is the value of n_{st} that satisfies the equation $t_{proc}(n_{st}) = t_{frame}$. By substitution, we get the following result:

$$n_{st,max} = \frac{f_{AP\ Clk}}{r_{frame} \cdot c}.$$

In our prototype we used an acquisition and processing clock with frequency $f_{AP\ Clk} = 15$ MHz, which is easily supported by the implemented VHDL code. With regard to the algorithm, the system has been designed to operate at a frame rate of $r_{frame} = 30$fps. At this rate, with two VILs being used, each person that passes through the door usually generates 4 to 6 state transitions. For our calculations, we introduce a value for $\bar{n}_{st} = 10$, that is, we consider the passage of two people in a row as an "average behavior".

By substituting these values into the previous formulas, we obtain $D = 0.3\%$ ans $n_{st,max} = 4167$. In order to reduce power consumption, it is critical that the duty cycle is as small as possible, so that most of the time the acquisition and processing clock is disabled even when the images need to be processed. If we assume the worst case of a state transition at every frame, a value of $n_{st,max} = 4167$ means that the algorithm fails after 138 s of continuous movement in the monitored area, i.e. there is no interval longer than t_{idle} in which no event is captured by the camera. In our algorithm, t_{idle} has been set to 2 s.

The obtained values tell that much more complex algorithms can be implemented on the proposed system.

In order to perform power measurements, we employed two Agilent 34411A digital multimeters. We set the one of them to operate as an ampermeter with range 10 mA and an integration time equal to 100 power line cycles, i.e. 2 s. With these settings, the instrument measures the voltage drop on a shunt resistance of 2 Ω. We measured the voltage drop on the device with the other multimeter, operating on a range of 10 V with an integration time of 100 power line cycles, and characterized by an input resistance greater than 10 GΩ. With this configuration the instruments normal mode noise rejection is maximized, and their loading effects are negligible with limited voltage drop due to the ammeter and current absorption by the voltmeter.

TABLE II
POWER CONSUMPTION

	$V\pm\Delta V$ [V]	$I\pm\Delta I$ [mA]
Video-sensor	3.28±0.02	1.73±0.09
FPGA core	1.22±0.01	0.44±0.02
FPGA I/O banks	3.28±0.02	0.34±0.02

(1) Idle Mode

	$V\pm\Delta V$ [V]	$I\pm\Delta I$ [mA]
Video-sensor	3.28±0.02	1.74±0.09
FPGA core	1.22±0.01	1.09±0.06
FPGA I/O banks	3.28±0.02	0.39±0.02

(2) Active Mode: only image acquisition

	$V\pm\Delta V$ [V]	$I\pm\Delta I$ [mA]
Video-sensor	3.28±0.02	1.74±0.09
FPGA core	1.22±0.01	1.15±0.06
FPGA I/O banks	3.28±0.02	0.41±0.02

(3) Active Mode: image acquisition and execution of the algorithm

In order to be able to monitor the behavior of the node, we programmed the FPGA to transmit the whole image to a host pc at every frame. We set the transmission rate through the RS-232 interface to 256000 bps. At this speed, the system could not support acquisition rates higher than 10 fps.

We do not expect a significant increment in the power consumption when the node will run at the frame rate which the algorithm has been designed for, since the time required for transmission forces the A/P Clock to be active for a longer period than the time spent to execute the algorithm.

The measurement results are shown in Tab. II. The values related to the video-sensor also comprise the consumption of other elements present on the board on which the device is mounted, such as trimmers and other resistors required for debugging. The consumption associated to the wireless module, which depends on the amount of data being transmitted and the poll rate, is neglected, but we can safely assume that its impact on the overall consumption is limited to some hundreds of μW, since we do not transmit any image, but just the number of people counted. Moreover, transmission does not need to take place at every frame, but it may be scheduled to be performed at regular intervals plus when some particular condition occurs, e.g. a large number of people are detected.

Assuming to power the node with two batteries with a capacity of 2200mAh at 3V, we can estimate the node's lifetime t_{life} as a function of the average waiting time between two following persons t_{wait}. Calculations were carried out considering the different voltages levels used to power the FPGA core, the I/O banks and the sensor. The results are reported in

TABLE III
LIFETIME ESTIMATION

t_{wait} [s]	$t_{sleep}^{\%}$	P [mW]	t_{life} [days]
2	0	8.44	32
4	0	8.37	33
8	50	7.88	35
16	75	7.64	36
100	96	7.43	37

Tab. III, where the parameter $t_{sleep\%}$ is the percentage of time in which the FPGA is not active. We calculated this value considering a time to cross the camera field of view $t_{cross} = 2\,\mathrm{s}$, so that $t_{sleep\%} = max\,\{0,\ (t_{wait} - t_{cross} - t_{idle})/t_{wait}\}$. The parameter P reported in Tab. refpower is the average power consumed by the entire node, calculated on the basis of the time spent in the several power configurations, given the period of the events t_{wait}. In the worst case, the node has to elaborate all the frames acquired in the time interval during which the person walks across the camera field of view, i.e. for a time interval of length t_{cross}. When the person has passed, it still works in Active mode, but without performing the algorithm since no state transition occurs. This condition hold for a time interval of length t_{idle}, after which the node enters the Idle mode until the next person enters the scene.

These results show that the efficient use of the low-power capabilities of system's components results in long lifetime. This is critical for video-surveillance applications, where the system should work for a long time without the need to replace its power source. It should be noted, though, that the lifetime depends on the rate of events to be monitored and therefore is application-specific.

V. CONCLUSION

In this paper we analyzed the performance of a wireless camera network node equipped with an ultra-low power, non-standard imager. The analysis revealed the effectiveness of the sensor for surveillance applications, showing that a battery-operated camera-based system can survive for a long period of time without replacement of the power source. This is possible by exploiting the low-power modes supported by devices such as microcontrollers and FPGAs.

REFERENCES

[1] N. Massari, M. Gottardi, S. Jawed, F.B. Kessler, and T. IRST. A 100μW 64×128 -Pixel Contrast-Based Asynchronous Binary Vision Sensor for Wireless Sensor Networks. In *IEEE International Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers*, pages 588–638, 2008.

[2] T. Teixeira, E. Culurciello, J.H. Park, D. Lymberopoulos, A. Barton-Sweeney, and A. Savvides. Address-event imagers for sensor networks: evaluation and modeling. In *Proceedings of the 5th international conference on Information processing in sensor networks*, page 466. ACM, 2006.

[3] T. Teixeira, D. Lymberopoulos, E. Culurciello, Y. Aloimonos, and A. Savvides. A lightweight camera sensor network operating on symbolic information. In *Proceedings of 1st Workshop on Distributed Smart Cameras, Held in Conjunction with ACM SenSys*. Citeseer, 2006.

[4] T. Teixeira and A. Savvides. Lightweight People Counting and Localizing for Easily Deployable Indoors WSNs. *IEEE Journal on Selected Topics in Signal Processing*, 2(4):493–502, 2008.

[5] Actel Corp. IGLOO Handbook. http://www.actel.com/, 2009.

[6] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan. Mesheye: a hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. In *IPSN07: Proceedings of the 6th international conference on Information processing in sensor networks*, pages 360–369. Citeseer, 2007.

[7] P. Chen, P. Ahammad, C. Boyer, S.I. Huang, L. Lin, E. Lobaton, M. Meingast, S. Oh, S. Wang, P. Yan, et al. CITRIC: A low-bandwidth wireless camera network platform. In *Proceedings of the International Conference on Distributed Smart Cameras*, 2008.

[8] M. Rahimi, R. Baer, O.I. Iroezi, J.C. Garcia, J. Warrior, D. Estrin, and M. Srivastava. Cyclops: in situ image sensing and interpretation in wireless sensor networks. In *Proceedings of the 3rd international conference on Embedded networked sensor systems*, pages 192–204. ACM, 2005.

[9] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128×128 $120dB$ $15\,\mu s$ latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid State Circuits*, 43(2):566–576, 2008.

[10] E. Viarani. Extraction of traffic information from images at DEIS. In *Proc. of the International Conference on Image Analysis and Processing*, pages 1073–1076, 1999.

[11] Q. Wang, M. Hempstead, and W. Yang. A realistic power consumption model for wireless sensor network devices. In *Proceedings of the Third Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 286 – 295, 2006.

[12] Texas Instruments. 2.4 Ghz IEEE 802.15. 4/ZigBee-ready RF transceiver. *Texas Instruments*, 2006.