An inexpensive wireless smart camera system for IoT applications based on an ARM Cortex-M7 microcontroller

M. Tresanchez*, A. Pujol, T. Pallejà, D. Martínez, E. Clotet, J. Palacín

Department of Computer Science and Industrial Engineering, University of Lleida, Lleida, Spain, 25001

Abstract

This paper presents a new proposal of a low-cost and low-power wireless smart camera device, designed to be used as an inexpensive smart image acquisition edge node for Internet of Things (IoT). Taking the advantages of the ARM Cortex-M7 microcontrollers, a high-performance STM32F7 microcontroller is used as a main digital signal processor (DSP) to acquire and transfer images at full rate, keeping the core idle to do real-time image processing. In order to design a very compact, low-cost and low-power wireless image sensor node, the DSP is supported by an embedded 1.3 Mega Pixel CMOS camera sensor and a low consumption 2.4GHz Wi-Fi module. This setup allows a custom build-in image processing algorithm for specific IoT sensing applications or a simply setup as low-cost streaming Motion JPEG (MJPEG) wireless camera node. In this work, both configurations have been tested and analyzed. Results show that the device can acquire and process images simultaneously at a full rate (30 fps) and the MJPEG transmission reaches 7.23 fps with a QVGA resolution.

Keywords: IoT node, Image sensor, Video streaming, ARM, Cortex-M7

1. Introduction

Image sensors are an important type of sensor in several emerging Internet of Things (IoT) applications. They provide detailed environment information using complex technology yet having a highly competitive market price due to their standardization. Their basic application as a camera system for monitoring or supervising a location through video streaming can be extended when they are combined with image processing techniques [1-4]. In IoT, image sensors may be applied in many fields, some recent contributions use image sensors for surveillance video systems in smart cities [5], for personalized healthcare systems, for recognizing and understanding human activities [6, 7] or for video streaming systems in vehicular IoT [8].

Image acquisition systems require a high performance computation in order to handle the large amount of data. Furthermore, if they have to process and transmit image information through a wide network, more powerful and expensive Digital Signal Processors (DSP) are needed and, consequently, the current consumption increases. This is an important drawback in IoT wireless sensors network management [9] where it is crucial to have low-cost and low-

power nodes because usually there are many and powered on all time. Additionally, another relevant factor to consider is the node size, weight and the hardware complexity for final product manufacturing. In [9] and [10] a more complex hardware is used based on System-On-Chip (SoC) which needs and operating system, making it less robust and requires a delicate assembly process that hinders its integration in a final IoT sensor product. In this work, an embedded image acquisition system based on the new high-performance Cortex-M7 microcontroller and a compact CMOS camera module is proposed as a smart wireless IoT image sensor node. This proposed device has a very low power consumption, ideal in IoT environments, and requires very few electronic components which help to achieve a small size product at a competitive production cost. In order to balance the performance and consumption, depending on the application level requirements, the selected hardware allows to configure different low-power modes. This is essential in outdoor IoT applications where the device uses external energy resources like solar energy [11]. In this work, the proposed system takes advantage of the microcontroller internal hardware peripherals to acquire and transfer images at the maximum wireless adapter throughput with a minimum computational load. This means that no software or operating system is required for the video

streaming process which leaves the CPU free to execute digital image processing for specific applications and thus, having an embedded smart device.

This paper is structured as follows. Section 2 gives an overview of the related works. Section 3 introduces the main electronic components of the proposed device. Section 4 defines the architecture configuration of the microcontroller for continuous image acquisition and transmission. Section 5 shows the performance results and discussion about the device's capabilities. Finally, section 6 presents the conclusions of this work.

2. Related work

Many different low-cost wireless vision system technologies and applications have been proposed in the literature. In [11], an outdoor wireless smart camera network powered by solar energy is presented. It uses the Raspberry Pi as main SoC processor system and a MSP430 microcontroller to adapt the consumption regarding the application requirement and battery life. In [12], an outdoor vison system for real-time dynamic object tracking and classification is proposed to improve the public security surveillance, handling correctly moving objects with occlusions. A low-cost and high accuracy smart video solution for elderly fall detection is presented in [13]. This paper proposes the use of a Pi Camera mounted on a Raspberry Pi as a low-cost solution. This system captures full HD (1920x1080) images via the raspivid and the simpleCV open source software. Similarly, Khoje *et al.* [14] present a smart video surveillance system for IoT, supported by a Raspberry Pi and a USB camera using OpenCV for detecting motion and human counting. An additional 8-bit ATmega microcontroller with a matrix keyboard and a GSM module gives extra features such as password validation, SMS notification, and appliances power off.

The use of the Field Programmable Logic Array (FPGA) to design a tailored smart vision system network for IoT is proposed in [15]. This approach defines a FPGA supported architecture to capture images and extract particular features. These features are received through RS232 by a microcontroller which transmits them to an IEE802.15.4 network. Birem *et al.* [16] design a modular smart camera architecture based on an Altera Cyclone-III EP3C120 FPGA and a 1.3 Mega Pixel CMOS image sensor. A build-in application for real-time image feature extraction is performed, and an additional CY7C68013 microcontroller is used for USB 2.0 communication. This camera system is upgraded with a Giga-Ethernet module in [17] proposing a camera network platform by means of the IEEE 802.15.4 protocol. Another alternative of an FPGA-based camera is presented by Fularz *et al.* [18]. They introduce an architecture based on a Xilinx Zynq-7000 SoC which allows easy and flexible image processing algorithm integration.

Although the recent high-performance microcontrollers cannot reach a performance as high as the FPGAs, they can be enough in applications that do not require high resolutions and fast frame rate, whereas the consumption and cost are critical. In [19], an embedded video system based on a high-performance ARM Cortex-M4 32-bit microcontroller and an Omnivision OV7670 CMOS camera is used to perform real-time red peach fruit detection and tracking. Due to the microcontroller memory limitations, a AL422B FIFO memory buffer is used to freeze the current 320x240 RGB565 frame and process it. The tracking system proposed in this work reaches 10 fps. Yang *et al.* [20] introduce the hardware design of a home wireless camera gateway for a ZigBee network. The hardware is supported by a STM32F103ZET6 ARM Cortex-M3 32-bit microcontroller, a 2 Mega Pixel OV2640 CMOS camera, and a 2.4GHz CC2530

Zigbee transceiver. The system takes a picture of 320x240 pixels and uploads it when smoke sensor exceeds a threshold. The same CMOS camera sensor was used in [21]. In this application, the camera module is configured to directly provide JPEG compressed images. Those images are processed by an STM32F103 ARM Cortex-M3 32-bit microcontroller and transmitted by a 2.4GHz nRF24LE1 transceiver. The objective of this work is to use this technology in a capsule endoscope. Results showed a maximum raw image transmission rate of 11 fps and a maximum transmission distance of 10 m.

3. Materials

3.1. Digital processor unit

The main processor unit used in this work is the STM32F746NGH6 [22] Micro Controller Unit (MCU) from STMicroelectronics which is based on the high-performance ARM Cortex-M7 32-bit RISC (Reduced Instruction Set Computing) core operating at up to 216 MHz clock speed. This MCU incorporates high-speed embedded memories: 1 Mbyte of Flash memory, 320 Kbytes of SRAM, 16 Kbytes of instruction TCM RAM (for critical real-time routines), 4 Kbytes of backup SRAM available in the lowest power modes, and an extensive range of enhanced I/Os and peripherals connected to an Advanced High-performance Bus (AHB) matrix (multi-AHB). It also has two general-purpose 8-stream DMA controllers with FIFOs and burst support, a configurable 8 to 14-bit parallel camera interface that can operate at up to 54 Mbps, and a flexible memory controller to access external memories. These features are essential to make the approach of this project possible.

According to the manufacturer's datasheet, the MCU has a maximum current consumption of 124 mA running at 210MHz with all peripherals enabled whereas in the lowest power mode (standby mode), without losing the program execution, is less than 3.6μA.

This work is carried out using the STM32F746G-DISCO advanced development board [22] (Fig. 1) based on this MCU. In addition, the board provides an external 128-Mbit SDRAM (64-Mbit accessible), used to store the temporal images during the data acquisitions, a 4.3-inch 480x272 LCD-TFT color display, used for camera feedback view, and a 30 Flexible Printed Circuit (FPC) connector, used to connect the external camera module.
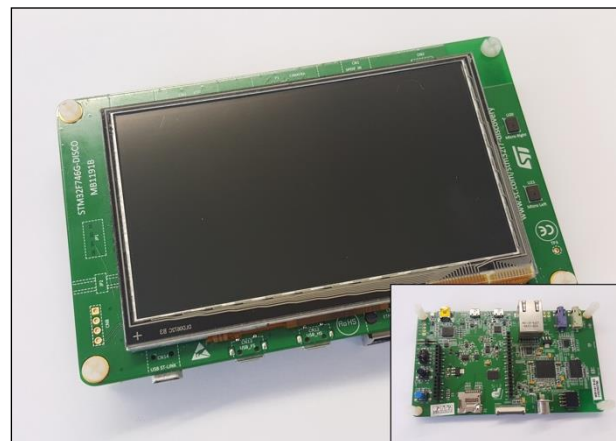

Fig. 1. STM32F746G-DISCO board [22]

3.2. Image sensor

The image sensor used is the Omnivision's OV9655 CMOS 1.3 Mega Pixel color sensor [23] which is integrated in a compact camera module (Fig. 2) along with an optical lens and a 24 pin FPC connector for communication. This image sensor has two communication buses: 1) a serial camera control bus (SCCB) to initialize and configure the parameters of the camera and 2) a parallel output pixel data bus up to 10 lines, which is continuously transmitting images. The output format is settable to Raw RGB, RGB (GRB 4:2:2, RGB 565, RGB 555), YUV (4:2:2) and YCbCr (4:2:2) formats. The sensor has an active 1280x1024 pixel array but the image size can be selected between SXGA, VGA, CIF, and any size scaling down from CIF to 40x30 by subsampling. All configurations have a maximum transfer rate of 30 frames per second except for SXGA which is 15 frames per second. Exposure, gain, white balance, band filter and other image parameters can be configured through the SCCB bus. In the same way, the image quality parameters such as color saturation, gamma, sharpness, lens correction, white pixel cancelling, noise cancelling, and 50/60 Hz luminance detection, can be configured as well.

The camera sensor works with different power supply sources. Its core needs 1.8V if the internal regulator is disabled. Its analog lines need a source from 2.45 to 3V that can be used for the IO lines. When the camera is in normal working conditions its current consumption is around 20mA whereas in standby mode it can reach as low as 1 μA.
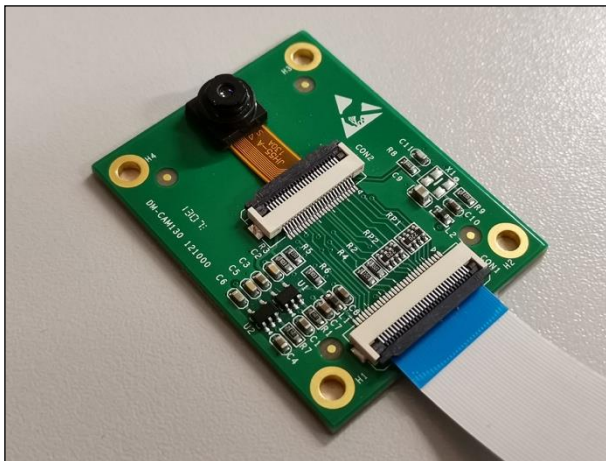

Fig. 2. OV9655 Camera module [23]

3.3. Wi-Fi module

The selected wireless connectivity for the proposed device is Wi-Fi technology which allows to connect the device into a TCP/IP local area network (LAN) with many other common IoT devices. The wireless module used is the ATWINC1500-MR210PB low-power consumption 2.4GHz 802.11b/g/n FCC Certified Wi-Fi from Microchip [24]. It has an internal dedicate MCU to handle the TCP/IP stack that allows to transfer information through a simple 48MHz Serial Port Interface (SPI). It enables to transmit data at up to 48 Mbps in 802.11g/n mode. This throughput makes it perfect for streaming video in low-cost and low-power embedded devices. Fig. 3 shows the ATWINC1500-MR210PB Wi-Fi module and the respective development extension board from the same manufacturer (ATWINC1500-XPRO).

The operation voltage of this wireless module ranges from 3.0V to 4.2V and the current consumption at 3.3V in 802.11n is

244mA for transmitting, 58.5mA for receiving and 380μA when the device is in Doze mode (the device is on but the communication is disabled). The module also includes a Power Amplifier (PA), Low-Noise Amplifier (LNA) and a Printed Circuit Board (PCB) antenna, giving a sensitivity up to -89dBm for receiving and 17.0dBm for transmitting in 802.11n mode. This fully integration helps to reduce the developing time and final manufacturing costs.
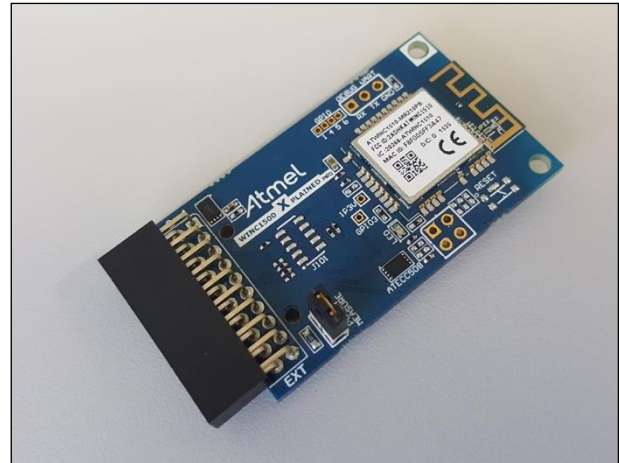

Fig. 3. ATWIN1500-XPRO Wi-Fi module [24]

4. Architecture configuration

4.1. Image acquisition

One of the main objectives of this setup is to achieve a high speed data acquisition keeping the CPU free. In order to do this, the MCU is designed with several hardware peripherals that can be configured to handle the different parts of the acquisition. The internal Digital Camera Interface (DCMI) is configured to capture images from the camera and store each pixel on a temporal peripheral register which is mapped by hardware into a memory region of the external SDRAM. This mapping is done using the Direct Memory Access (DMA) that provides high speed block data transfer by hardware between peripherals and memories, and the Flexible Memory Controller (FMC) to translate bus transactions of an external memory to internal memory registers. Fig. 4 shows the hardware architecture diagram used for image acquisition.

The image sensor OV9655 is configured to gather the data in RGB565 mode (each pixel consists of 16-bits) and send it through a 30 pin FPC as a constant stream running at 30 frames per second. The data is sent one byte at a time in an 8-bit parallel line together with the Horizontal Synchronization (HSYNC), Vertical Synchronization (VSYNC) and the Pixel Clock (PCLK) signals, and is received by the DCMI, where it is packed into a 32-bit register called Data Register (DR).

A DMA request is generated each time the camera interface receives a complete 32-bit data block in its data register and it is sent to the DMA2, which initiates a transfer from the DR to an internal First-In First-Out (FIFO) buffer through the AHB peripheral bus. When the threshold level of the FIFO is reached the contents are drained and stored into the destination through the AHB memory bus. The data is handled by the FMC in order to translate AHB transactions in the appropriate external SDRAM protocol, then it is transferred in 16-bit blocks and stored into a 12-bit memory address.
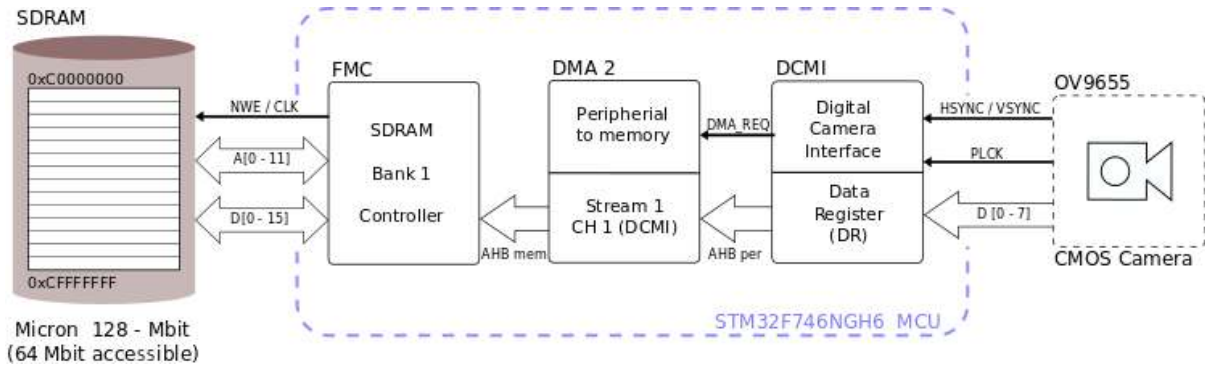
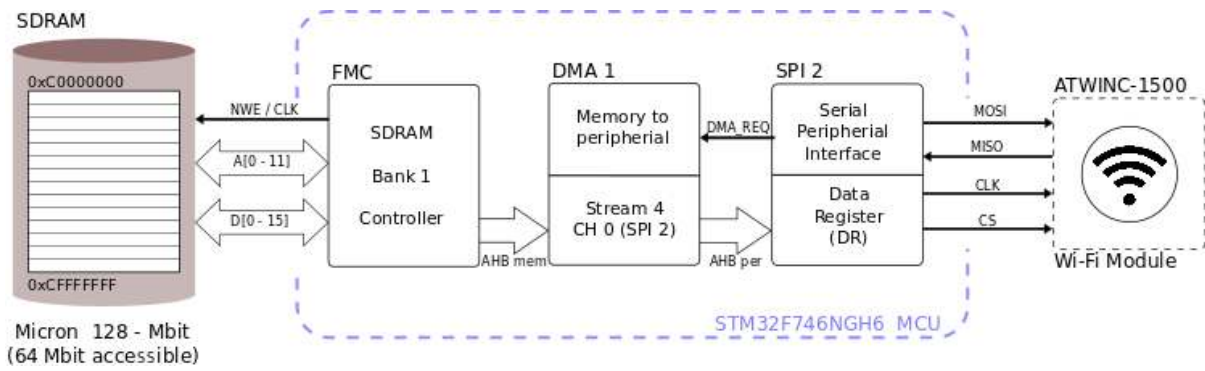**Fig. 4. Proposed hardware architecture for image acquisition**

Fig. 5. Proposed hardware architecture for the wireless communication

4.2. Wi-Fi transmission

The proposed device is designed to be capable to capture and process each frame as fast as possible and transmit the extracted information via Wi-Fi, whether this information is simple data or a full image. The Wi-Fi module is able to transmit up to 48Mbps. This is enough to transmit images with basic QVGA resolutions as long as the MCU hardware guarantees the continuous data acquisition. Fig. 5 shows the hardware architecture configuration to transmit any data stored in the external SDRAM to the Wi-Fi module using the internal DMA and the SPI2 peripheral. The goal of this scheme is to reach a maximum SPI throughput rate keeping the CPU idle for possible image processing tasks.

In the case of transmitting (Fig.5), the reverse method to the image acquisition is applied: the image data is taken from the external SDRAM using an FMC, and the DMA1 creates a memory map from the AHB memory to the DR of the SPI2 peripheral in order to transfer the entire image data to the Wi-Fi module in one DMA transaction. The SPI2 peripheral of the MCU acts as a master whereas the dedicated MCU of the Wi-Fi module act as a slave which translates the data received to TCP/IP frames using its internal software stack. In the case of receiving mode, due to the small amount of received data, all the incoming information is just handled with an interrupt byte by byte.

5. Results and discussion

The proposed device and hardware architecture has been tested and some performance experiments have been carried out with three different camera resolutions, 160x120, 320x240 and 640x480. Table 1 shows the device performance acquiring and storing images in real-time at these resolutions. The PCLK value depends on the resolution to maintain the maximum frame rate of the camera and the capture time of a horizontal line, HSYNC, also it is proportional to the resolution. After every frame a time gap of 1.33 ms is generated regardless of the resolution. This time is considered an idle time for the acquiring system, which has already finished transmitting the current frame and has to wait for the next one. Considering these acquisition times, the measured frame rate matches the manufacturer's datasheet, which specifies a maximum of 30 fps for all tested resolutions.

Table 1.   Capturing performance

Camera Resolution	PCLK (MHz)	HSYNC capturing time (µs)	Idle time between frames (µs)	Reached frame rate (fps)
160 x 120 (QQVGA)	6.00	266.68	1331.68	29.88
320 x 240 (QVGA)	12.00	133.32	1342.82	29.99
640 x 480 (VGA)	24.00	66.67	1336.70	30.00

Moreover, the capturing tests corroborate that the MCU allows to perform all the data gathering and storing at full rate leaving the main CPU with almost no load. This is possible thanks to the DMAs interfaces, used in almost all parts of the proposed architecture, which are designed to move blocks of data very fast by hardware. Thus, the CPU is free during the images transmission and it can be used to execute image processing tasks, keeping the same frame rate. This proposal requires a double buffer algorithm to swap the memory area at each frame in order to have access and process an entire image while the next one is being acquired.

In order to test the image acquisition and Wi-Fi communication performance, the proposed sensor node has been configured to work as a streaming Motion JPEG (MJPEG) server (Figure 6). Two open source libraries have been used for JPEG compression and TCP/IP protocol management. The LibJPEG [25] which is a C based software tool that allows to encode a single image from a bitmap (*.bmp*) file to a JPEG (*.jpg*) file with variable size and selectable compression quality, and the Lightweight IP (*LwIP*) library [26] which is a tiny stack implementation of the TCP/IP protocol designed to reduce resources usage while still having the full TCP. This makes it perfect for embedded devices with a small amount of RAM memory.

Figure 6 shows the configuration setup tested in this project. The MCU is acquiring RGB565 images and encoding them using the LibJPEG. The JPEG images are sequentially transmitted by Wi-Fi using the LwIP driver which handles an TCP HTTP MJPEG server. The ATWINC-1500 Wi-Fi module and the smartphone are connected to the same 2.4GHz 802.11n wireless access point. In the picture, the smartphone is receiving JPEG images continuously in response of a HTTP GET query (http://192.168.0.40/video.mjpg). At the same time, the microcontroller shows the images acquired and the transmission performances on the LCD. In this setup, the image acquisition has been carried out using double buffer swapping in order to start the JPEG compression of the current frame while the next is being acquired.
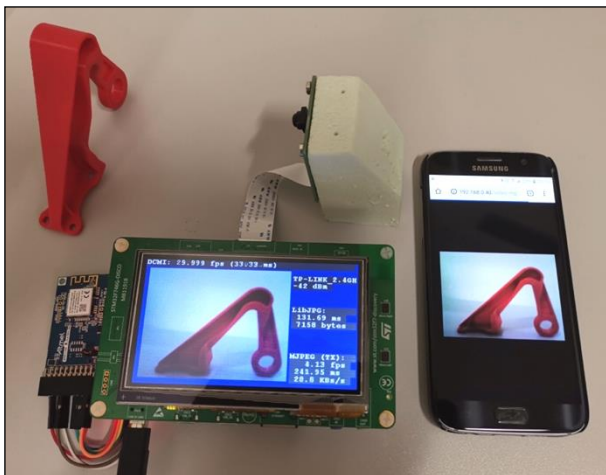

Fig. 6. Wireless camera sensor system proposed streaming JPEG images (MJPEG) to a smartphone

Figure 7 shows the results of the frames per second reached using a stable Wi-Fi signal for different resolutions and JPEG compression rates. The results show that the image quality compression is more relevant at higher rates and therefore an optimal point when comparing image quality and framerate should be between 60 and 80% of compression rate. The best frame rate obtained was 7.23 fps with a 25% compression rate and an image resolution of 160x120. It was very low regarding

the frame rate of DCMI image acquisition (30 fps), this is mainly because both JPEG compression and TCP/IP stack control are done by software and they take more time to compress (34.67 ms averaged for 160x120) and transfer (121.03 ms averaged for 160 x 120) than the acquisition (33.33 ms). In case that the JPEG streaming was the final application, a CMOS sensor supporting a JPEG output format, such as the Omnivision's OV2640 [20, 21] sensor, or a microcontroller designed with a hardware JPEG codec peripheral [27], would improve these results.
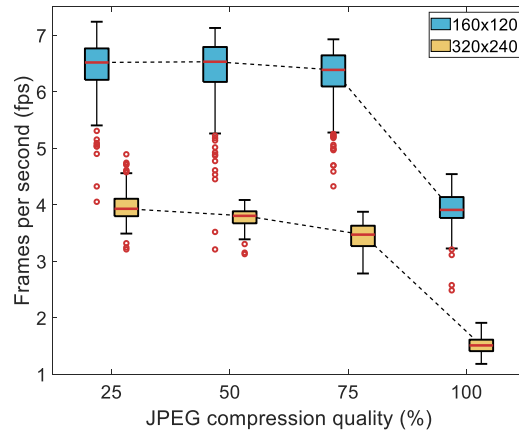

Fig. 7. Frames per second reached for different resolutions and compressions quality

Figure 8 shows the MJPEG throughput of 160 x 120 image streaming with a compression rate of 75% with different signal strength: -20, -40, -60 and -80 dBm. When the strength signal is between -40 and -60 dBm, the transmission is almost stable obtaining a MJPEG throughput of 13 Kbytes/s. When the signal strength is too high (distance to the access point less than 1 m), the transmission begins to be unstable and, from a strength of -80 dBm (17 m), the throughput decreases drastically below 5 Kbytes/s.
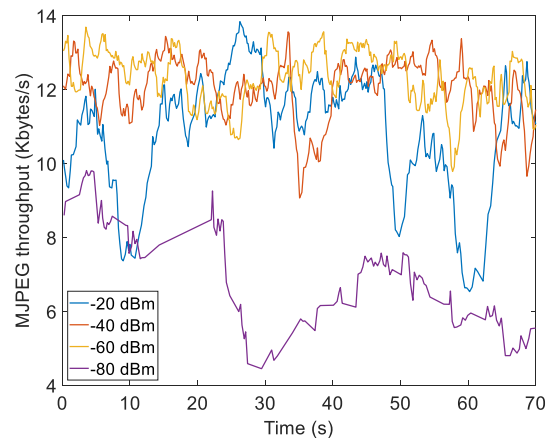

Fig. 8. MJPEG throughput for different wireless signal strengths

Table 2 shows the performance comparison of streaming MJPEG images with and without a previous build-in image processing. The averaged time of streaming original images is compared with the averaged time of streaming processed images. The algorithm tested consists on a basic red segmentation and a centroid location (Fig. 9). It cumulates the coordinates of the pixels that reach the red threshold and then calculates the centroid by averaging them. In the case of processed images, the device transmits the images with the result

of the segmentation (black and white pixels) indicating the location of the centroid by red color (Fig. 9-b). The resolution was 320x240 pixels with 75% of JPEG compression rate. As shown in Table 2, the image throughput with image processing is proportional to the time required for processing. This is because all free CPU time between DCMI image acquisitions is already taken by the image compression (LibJPEG) and transmission (LwIP) libraries.

Table 2.   Comparative performance streaming original images and build-in processed images

MJPEG Output	DCMI capture rate (fps)	Image processing time (ms)	JPEG compression time (ms)	Total transmission time (ms)
Original images	30.00	-	140.38	288.01 (3.47 fps)
Processed images	30.00	103.56	142.57	394.03 (2.53 fps)

Fig. 9.   Intercepted MJPEG streaming images: a) original images, b) modified build-in processed images

In order to evaluate the image acquisition and transmission quality, 200 MJPEG frames (original 320x240 JPEG images with 75% compression rate) were streamed to the destination host and recorded in a file. To be sure all frames were as similar as possible, the camera end was placed in a controlled scenario with uniform LED light, focusing on a static object. The first recorded frame, I_{ref}, was set as reference and subtracted from the rest. The average absolute differences in percentage (eq. 1) were used to detect glitches or irregularities in the video transmission. Figure 10 shows that the average of the differences was 0.515% due to the camera auto adjust settings. An ocular inspection to

the extracted images of the recorded MJPEG also corroborates the video streaming good performance.
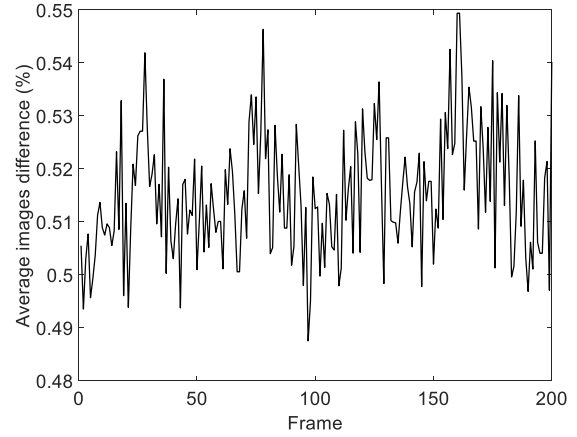
Fig. 10. Average difference between streamed images

$$AAD_i = 100 \cdot mean\left|I_{ref} - I_i\right|, i \in [1,199] \qquad (1)$$

Finally, the proposed wireless video system is compared in terms of cost and power performance against other two shelf hardware solutions. The first one is the Raspberry Pi Zero W [28], which is based on a SoC Broadcom BCM2835 with an ARM11 CPU running at 1GHz and an internal 512 Mbytes of RAM, along with the Raspberry Pi camera module V2.1 that has a Sony IMX219 8 Mega Pixel sensor which supports up to 1080p (1920×1080) at 30 fps. This system is configured to work as a 2.4GHz 802.11n wireless Real Time Messaging Protocol (RTMP) streaming server using the FFmpeg GNU Software [29]. The second solution, is an off the shelf wireless IP-surveillance commercial camera, the Edimax IC-3115W [30]. This camera has an integrated image sensor of 1.3 Mega Pixel with a maximum resolution of SXVGA (1280 x 960) at 30fps and a 2.4GHz 802.11n wireless network connection that allows to get the sensor data through streaming MJPEG.

Table 3.   Price and power consumption of three types of wireless IP camera solutions

	Approximated cost (€)	Power at full rate streaming (mW)		
		Minimum	Maximum	Average
Our proposed device	**26.30**	**686.44**	**710.02**	**703.65**
STM32F746NGH6 MCU	*12.55*	*286.89*	*289.52*	*288.20*
OV9655 camera	*6.25*	*129.89*	*130.54*	*130.22*
ATWINC1500 Wi-Fi module	*7.50*	*269.66*	*289.95*	*285.23*
Raspberry Pi SoC solution	**43.90**	**1022.95**	**1257.48**	**1087.82**
RPi Zero WH board	*16.95*	*934.35*	*1168.24*	*997.94*
RPi Camera V2.1	*26.95*	*88.60*	*89.24*	*89.88*
Edimax IC-3115W IP camera	**22.40**	**1781.43**	**2100.79**	**1966.06**

Table 3 shows a comparison of these three types of solutions in terms of cost and power consumption at a maximum speed of wireless streaming. The Edimax IC-3115W is the most economical because it is an end-product and has a competitive price. In contrast, this solution has the worst power consumption

which reaches up to 2100 mW. The Raspberry Pi has better image sensor with better power consumption regarding to our selected sensor, however this affects its price which is higher than the processor itself. In the case of the proposed device, the cost only considers the three main electronic components

6

(MCU, image sensor and Wi-Fi module) because the development platform has many unused components. The approximated cost would not be much different when manufacturing a custom design and it is feasible due to the few required electronic components.

Results show that the proposed device is the best solution in terms of energy consumption but does not differ much compared to the Raspberry Pi Zero W solution, which is more powerful and captures better quality images. One advantage of our proposal is that the MCU has many possibilities to work in low-power modes. Thus, if the final application does not require to be active all the time, it would be possible to reduce its consumption significantly.

6. Conclusions

In this work, a new alternative of wireless image camera node for IoT applications was proposed. The device was designed using the novelty ARM Cortex-M7 microcontrollers along with low-cost and low-power 1.3 Mega Pixel CMOS sensor and 2.4 GHz Wi-Fi module. An architecture configuration was proposed in order to achieve the lowest CPU use and the maximum camera frame rate. Performance tests showed that during the acquisition and wireless transmission, all CPU time between frames (33.3 ms) can be used for real-time image processing without dropping any frame.

The proposed hardware architecture is as good as a complex SoC based vision system, as long as the CPU speed reduction does not increase the image processing time by more than 33.3 ms. In addition, the proposed solution is more economical and consumes less power than the standard ARM-based SoC systems, like the Raspberry Pi Zero W, and also, can be more robust since it works without operating system.

The wireless transmission results show that the proposed solution is not well fitted for image streaming due to the required time for JPEG software compression and the low transmission rate reached with the MJPEG format (a maximum of 7.23 fps for QVGA). Therefore, the main IoT application of this device should be to process images internally and transmit by Wi-Fi only the extracted information, and occasionally to send the entire frame if it is required.

References

[1] Mathe K, Busoniu L. Vision and Control for UAVs: A Survey of General Methods and of Inexpensive Platforms for Infrastructure Inspection. Sensors 2015; 5:14887-14916. https://doi.org/10.3390/s150714887

[2] Liu H, Zhou Q, Yang J, Jiang T, Liu Z, Li J. Intelligent Luminance Control of Lighting Systems Based on Imaging Sensor Feedback. Sensors 2017; 17(2): 321. https://doi.org/10.3390/s17020321

[3] Sarikana SS, Ozbayoglub AM, Zilci O. Automated Vehicle Classification with Image Processing and Computational Intelligence. Procedia Computer Science 2017; 114:515-522. https://doi.org/10.1016/j.procs.2017.09.022

[4] Font D, Moreno J, Tresanchez M, Teixido M, Palleja T, Palacin J. Compact and low cost embedded vision system for fruit detection and tracking. 9th European Conference on Precision Agriculture (ECPA 2013); 2013, Lleida, Spain.

[5] Tian L, Wang H, Zhou Y, Peng C. Video big data in smart city: Background construction and optimization for surveillance video processing. Future Generation Computer Systems 2018. https://doi.org/10.1016/j.future.2017.12.065

[6] Qia J, Yang P, Min G, Amft O, Dong F, Xu L. Advanced internet of things for personalised healthcare systems: A survey. Pervasive and Mobile Computing 2017; 41:131-149. https://doi.org/10.1016/j.pmcj.2017.06.018

[7] Joshi NB, Nalbalwar SL. A fall detection and alert system for an elderly using computer vision and Internet of Things. IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT); 2017, pp. 1276-1281, Italy. https://doi.org/10.1109/RTEICT.2017.8256804

[8] Aliyu A, Abdullah AH, Kaiwartya O, Cao Y, Lloret J, Aslam M, Joda UM. Towards video streaming in IoT Environments: Vehicular communication perspective. Computer Communications 2018; 118:93-119. https://doi.org/10.1016/j.comcom.2017.10.003

[9] Shafique Ahmad Chaudhry, Weiping Song, Muhammad Habeeb Vulla, Cormac Sreenan. EMP: A Protocol for IP-Based Wireless Sensor Networks Management. Journal of Ubiquitous Systems and Pervasive Networks 2011, Vol. 2 pp. 15-22 https://doi.org/10.5383/JUSPN.02.01.002

[10] Othman NA, Aydin I. A new IoT combined body detection of people by using computer vision for security application. International Conference on Computational Intelligence and Communication Networks (CzCN); 2017, Girne, Cyprus. https://doi.org/10.1109/CICN.2017.8319366

[11] Abas K, Obraczka K, Miller L. Solar-powered, wireless smart camera network: An IoT solution for outdoor video monitoring. Computer Communications 2018, 118:217-233. https://doi.org/10.1016/j.comcom.2018.01.007

[12] Johnsen S, Tews A. Real-time object tracking and classification using a static camera. International Conference on Robotics and Automation (ICRA), workshop on People Detection and Tracking; 2009, Japan.

[13] Waheed SA, Khader PSA. A Novel Approach for Smart and Cost Effective IoT Based Elderly Fall Detection System Using Pi Camera. IEEE International Conference on Computational Intelligence and Computing Research (ICCIC); 2017, pp. 1-4, Tamil Nadu, India. https://doi.org/10.1109/ICCIC.2017.8524486

[14] Gulve SS, KhoJe SA, Pardeshi P. Implementation of IoT-Based Smart Video Surveillance System. International Conference on Computational Intelligence in Data Mining (CIDM); 2016, pp. 771-780, Barcelona, Spain. https://doi.org/10.1007/978-981-10-3874-7_73

[15] Maggiani L, Iodice GM, Gassani A, Salvadori C, Azzarà A, Saletti R, Pagano P. A novel architecture of a Smart Camera Networks tailored to the IoT. Workshop on Architecture of Smart Camera; 2013, pp. 1-38, Sevilla, Spain.

[16] Birem M, Berry F. DreamCam: A modular FPGA-based smart camera architecture. Journal of Systems Architecture 2014, 60:519-527. https://doi.org/10.1016/j.sysarc.2014.01.006

[17] Bourrasset C, Maggiani L, Sérot J, Berry F, Pagano P. DreamCAM: A FPGA-based platform for smart camera networks. International Conference on Distributed Smart

Cameras (ICDSC); 2013, Trento, Italia. https://doi.org/10.1109/ICDSC.2013.6778244

[18] Fularz M, Kraft M, Schmidt A, Kasínski A. The Architecture of an Embedded Smart Camera for Intelligent Inspection and Surveillance. Progress in Automation, Robotics and Measuring Techniques: Control and Automation 2015, 43-52. https://doi.org/10.1007/978-3-319-15796-2_5

[19] Teixidó M, Font D, Pallejà T, Tresanchez M, Nogués M, Palacín J. An Embedded Real-Time Red Peach Detection System Based on an OV7670 Camera, ARM Cortex-M4 Processor and 3D Look-Up Table. Sensors 2012; 12:14129-14143. https://doi.org/10.3390/s121014129

[20] Yang B. Design and Implementation of Intelligent Home Wireless Gateway Based on STM32. International Conference on Information Science and Control Engineering; 2017, pp. 258-260, Changsha, China. https://doi.org/10.1109/ICISCE.2017.62

[21] Ya'nan Z, Guijin X, Jiansheng X. The Wireless Image Transmission System of Capsule Endoscope Based on STM32F103. International Conference on Mechanical, Electronic and Information Technology Engineering (ICMITE); 2016, pp. 297-303, Chongqin, China. https://doi.org/10.12783/dtetr/ICMITE20162016/4613

[22] STM32F746NGH6 STMicroelectronics ARM Cortex-M7 Microcontrollers. http://www.st.com/en/microcontrollers/stm32f7-series.html

[23] OV9655 Omnivision 1.3 Mpx CMOS color sensor. http://www.ovt.com

[24] ATWINC1500-MR210PB 2.4GHz 802.11b/g/n Microchip Wi-Fi adapter. http://www.microchip.com/wwwproducts/en/ATwinc150

[25] LibJPEG Open source driver for JPEG coding and encoding. http://libjpeg.sourceforge.net/

[26] Lightweight IP Open source driver for TCP/IP stack control. http://savannah.nongnu.org/projects/lwip/

[27] STMicroelectronics. Application note AN4996: Hardware JPEG codec peripheral in STM32F76x microcontrollers. https://www.st.com/resource/en/application_note/dm00356635.pdf

[28] Raspberry Pi Zero WH single-board computer. http://www.raspberrypi.org/

[29] FFmpeg free software cross-platform solution. http://www.ffmpeg.org/

[30] Edimax IC-3115W wireless network IP-surveillance camera. http://www.edimax.com/