Experimental Evaluation of a Video Streaming System for Wireless Multimedia Sensor Networks

Stefano Paniga, Luca Borsani, Alessandro Redondi, Marco Tagliasacchi, Matteo Cesana

Dipartimento di Elettronica e Informazione,
Politecnico di Milano,
P.zza Leonardo da Vinci, 32
Milano, Italy
{paniga, borsani, redondi, tagliasacchi, cesana}@elet.polimi.it

***Abstract*—Wireless Multimedia Sensor Networks (WMSNs) are recently emerging as an extension to traditional scalar wireless sensor networks, with the distinctive feature of supporting the acquisition and delivery of multimedia content such as audio, images and video. In this paper, a complete framework is proposed and developed for streaming video flows in WMSNs. Such framework is designed in a cross-layer fashion with three main building blocks: (i) a hybrid DPCM/DCT encoder; (ii) a congestion control mechanism and (iii) a selective priority automatic request mechanism at the MAC layer. The system has been implemented on the IntelMote2 platform operated by TinyOS and thoroughly evaluated through testbed experiments on multi-hop WMSNs. The source code of the whole system is publicly available to enable reproducible research.**

I. Introduction

The integration of low-power wireless networking technologies such as Wireless Sensor Networks (WSN) with inexpensive complementary metal-oxide semiconductors (CMOS) cameras and microphones has enabled the development of the so called Wireless Multimedia Sensor Networks (WMSNs), that is, networks of wireless devices capable of sensing multimedia content such as audio, still images and video, as well as scalar sensor data from the environment. WMSNs may enable new applications ranging from enhanced surveillance and monitoring systems, to advanced services for health care and telemedicine, where WMSNs can be integrated with real-time localization systems such as the one in [1].

Wireless multimedia sensor networks are in many ways unique and more challenging with respect to traditional sensor networks: (i) sensor devices are constrained in terms of battery, memory and computational capability while multimedia processing requires to store, process and transmit large amount of data; (ii) applications of multimedia sensor networks require real-time data from camera networks: on the one hand these requirements struggle with the network side limitations (variable channel capacity, limited bandwidth, etc...), and on the other hand with the constraints imposed by the limited energy resources and processing capabilities of available embedded processors for sensor nodes.

Different valuable surveys such as [2], [3] and [4] clearly highlight the state of the art and the main research challenges for the development of WMSNs. Several aspects of WMSNs are discussed, including application scenarios, existing solutions and open research issues at the different layers of the communication stack (application, transport, network and medium access control), cross-layer optimizations and streaming mechanisms. The main points which are made clear in these surveys can be summarized in what follows:

- all the layers of the communication stack need to be re-adapted since most of the existing algorithms and protocols originally developed for traditional wireless sensor networks are not suited to support multimedia communication;
- cross-layered design approaches are preferable to minimize latency thus guarantee the application-specific Quality of Experience, and further reducing protocol-overhead;
- in the signal processing area, multimedia encoding techniques need to be efficiently applied in order to achieve high coding efficiency and robustness.
- Implementations on commercial hardware are preferable, in order to test effectively the performance of the proposed techniques throughout extensive experiments on such testbeds.

In this work, we present a video streaming system for wireless multimedia sensor networks. The proposed system features a framework for multimedia delivery (still images and video) built on top of a multi-hop wireless sensor network. A hybrid DPCM/DCT coding scheme is implemented to achieve high compression while maintaining perceptual video quality, together with a multi-hop congestion control system that minimizes latency due to buffer overflows in intermediate nodes. The system is implemented using the IntelMote2 platform with a IMB400 board from Crossbow that adds multimedia capability. System firmware is written in the NesC language and based on the TinyOS operating system. In order to enable reproducible research on video streaming for WMSN, we made our source code publicly available at [5].

The remainder of this paper is organized as follows. In Section II we review the state of the art on video streaming in WMSN. In Sections III and IV we describe in details our solution. In Section V we evaluate our system through extensive experiments and finally in Section VI we conclude the paper.

II. Related work

In recent years the field of WMSNs has received much attention in the networking research community and as an interdisciplinary field of interest. Several efforts have been made to achieve important results in various fields related to WMSN, from the research on specialized hardware to the development of efficient algorithms and protocols for multimedia transmission. Image transmission over low-bitrate networks such IEEE 802.15.4 and Zigbee is addressed in [6], where both JPEG and JPEG2000 compression schemes are tested, highlighting limitations of the network and evaluating both Peak Signal to Noise Ratio (PSNR) and byte error rate. The work in

978-14577-0900-5/11/$26.00 ©2011 IEEE

[7] describes a simple single hop network architecture based on the Fleck3 platform, which enables to acquire, compress and send to a base node QVGA images using a compression technique very similar to JPEG. The system is evaluated in details with experiments focused on the achieved PSNR and energy usage.

For what concerns video transmission, [8] investigates issues associated with the transport of multimedia streams across WSNs, introducing a flow-control algorithm based on pipelined transmission to increase network performance. In [9], the authors propose a cross-layer approach to dynamically tune the rate of video transmissions to minimize distortion while achieving fairness among multiple concurrent video flows in multi-hop WSNs.

Chen et al. in [10] propose a real-time video surveillance system composed of two IP-cameras and sixteen low-cost wireless sensors in a multiple-tier architecture. In this case the sensor network can detect and track an object and wake up the IP cameras to record these events. Other similar multiple-tier architectures can be found in [11], [12] and [13].

Politis et al. in [14] propose a scheduling algorithm for transmitting video packets over multi-paths according to their importance (high priority packets over high bandwidth paths), including a power-aware packet scheduling mechanism that selectively drops the least significant video packets prior to transmission in order to save energy. Guo and Little in [15] propose a QoS-enabled dynamic path formation algorithm to yield throughput-aware video delivery over WSNs by distributing a limited number of mobile sinks to the bottleneck location for each video stream. A multi-path multi-stream video delivery architecture for WMSNs is proposed in [16]: the video sequence is encoded in multiple streams and each of these is assigned to a different path for ensuring load balancing and error resilience. As the packets traverse the multi-hop network, they are partially and progressively decoded through a distributed error recovery framework to recover from channel-induced errors.

The work in [17] describes a multi-channel approach for video-forwarding in WSN, providing the rationale for employing multiple channels in the transmission of constrained video feeds and describing the practical implementation of the proposed approach in commercially-available sensor network hardware.

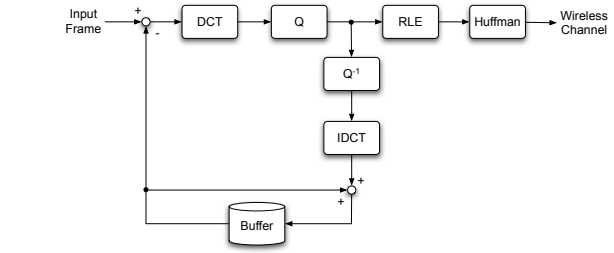
Although the research community in video sensor networks is very vast and active, many of the proposed solutions are evaluated through simulations due to the lack of an accessible video system implementation running on commercial hardware such as the IntelMote2 platform. The main contribution of this work is the implementation and extensive evaluation of a multi-hop streaming video system based on the standard development framework of TinyOS. We encourage the use of the publicly available source code to perform evaluation of other protocols and encoding techniques in real testbeds.

III. Video System

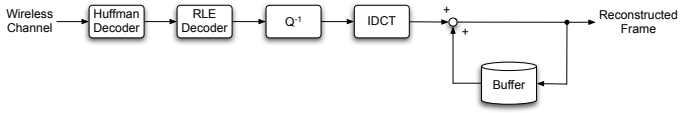
This section describes in details the implementation of the video streaming system at the different layers of the communication stack.

A. Hardware platform

The underlying hardware platform is composed of Imote2 motes by Intel, which are built around an integrated wireless micro controller consisting of the low-power 32-bit PXA271 Marvell processor. They include 256 KByte SRAM, 32 MByte Flash memory, 32 MByte SDRAM and several I/O options. Radio communication is enabled by the 802.15.4-compliant Texas Instruments/Chipcon CC2420, that supports a 250 Kbps data rate in the 2.4 GHz band. Multimedia capability is added using the Crossbow IMB400 module, which includes a OmniVision OV7670 Camera Chip, an audio capture and playback codec (Wolfson WM8940) as well as a Panasonic PIR motion sensor.



(a)



(b)

Fig. 1. Block diagram of the implemented Differential Motion JPEG video codec. (a) Encoder: the input frame is DCT transformed and resulting coefficients are quantized according to a specified quality factor. The resulting JPEG coded frame is reconstructed and stored in a buffer for subsequent DPCM encoding. Before transmission, the resulting bitstream is entropy encoded using run length coding and Huffman coding. (b) Decoder: input bitstream is decoded and added to previously reconstructed frames.

B. Application Layer

We describe the video system starting from the application layer and the multimedia encoding technique adopted. The main design objectives of a multimedia coder for WMSNs are high compression efficiency together with low complexity and error resiliency.

In the traditional broadcasting paradigm, a video sequence is compressed once with a complex encoder and decoded several times with simpler decoders. Standard encoders such as the ones in MPEG or H.264 rely on demanding processing algorithms and may not be supported by sensor networks, which are characterized by nodes with low processing power and limited energy budgets. Hence, the aforementioned paradigm may be unfeasible for WMSNs and encoding techniques that move the processing burden to the decoder at the sink, like distributed video coding [18], seem to be promising.

However, we decided to adopt a simple yet effective hybrid DPCM/DCT coding scheme to achieve an acceptable compression gain while keeping the computational complexity low. With reference to Figure 1(a) the first frame acquired from the CMOS camera is processed by the sensor node to produce a JPEG coded I-frame (in this case the buffer is empty). We used the standard JPEG baseline with quantization of DCT coefficients, followed by run length coding and Huffman coding. For subsequent frames, only the difference with the previous frame is encoded (resulting in a so called P-frame), thus producing a compressed bitstream. At the decoder side (as shown in Figure 1(b)), the received frames are stored in a buffer and summed to subsequent prediction residuals to reconstruct the original sequence. In our implementation the video encoder produces sequences of one I-frame followed by nineteen P-frames, achieving an average compression ratio of 50:1 with respect to raw frame transmission and 2:1 compared to a simple encoder that codes each frame separately with JPEG and send it on the radio[1].

[1]Compression ratios are computed for a QVGA video of a scene with different type of motions and a target PSNR of 38 dB

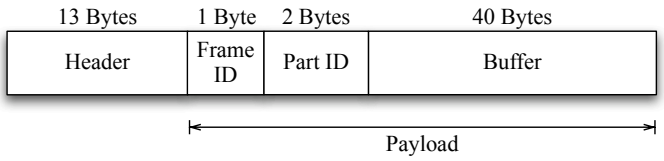


Fig. 2. Video packet format

For what concerns frame acquisition and processing time, typical results for a QVGA frame are about 90 ms for the acquisition process and 270 ms for the DPCM compression stage, thus an upper bound on the frame rate for our system is about 2.8 fps for a QVGA compressed video.

C. Transport, Network and MAC Layer

Two main objectives are of great importance in the design of suitable network protocols for data delivery in WMSNs: *timeliness* and *reliability*. The former is usually of greater concern than the latter in multimedia applications, since real-time is often required; connectionless transport protocols based on UDP may thus be a solution. However the data packets generated by compression technique such as the one illustrated in Section III-B, have different priority and cannot be indiscriminately dropped: I-frames carry content that cannot be easily concealed when lost and their transport must be secured. Thus, some form of *reliability* must be introduced for these packets, as well as a congestion control mechanism that adapts the packet generation rate of the source to the current level of congestion in the network. These two features, reliability and congestion control, are typical of TCP-based protocols: hence no single transport protocol exists that addresses the diverse concerns of WMSNs.

Leveraging these observations, we implemented the following features:

- **Selective reliability:** With reference to Figure 2, the output bitstream from the encoder is divided in packets of 56 bytes. The first 13 bytes contains the TinyOS MAC header, the field PartID (2 bytes) contains the sequence number of the packet within the current frame, and the field FrameID (1 byte) contains the identifier of the current frame (for I-frame the identifier is always 0). Both source nodes and forwarding nodes in the network can thus check the FrameID field and request acknowledgments only for video packets belonging to I-frames. A retransmission mechanism together with duplicate packet dropping is also implemented.
- **Congestion control mechanism:** the wireless medium is inherently unreliable and the capacity of each link in the network is location-dependent, varies continuously and may be bursty. The reception rate of packets for a particular node can be far greater than the forwarding rate: this fact, coupled with the limited memory size of sensor nodes, causes packets dropping due to buffer overflows. We implemented an explicit and recursive congestion control mechanism to cope with this situation.
  Each node has two thresholds called respectively *stop threshold* and *restart threshold* that allow to know if the buffer is near saturation and to take the proper countermeasures. When the incoming messages buffer size is greater than the *stop threshold*, the node continues to accept new packets but sends an explicit control message to its source node. Upon reception of this message the source node stops to send packets and remains in that state until it does not receive another control message from its destination node meaning that it can restart sending packets. The latter message is sent by the saturated node only when its buffer size decreases under the *restart threshold*. When a node is in the stop state it will fill rapidly its own buffer and will ask its source node to stop sending messages. This stop condition will be propagated recursively until the camera node stops to inject new packet into the network, thus reducing congestion.

For what concerns the MAC layer, the proposed solution relies on the standard IEEE 802.15.4 CSMA protocol with clear channel assessment (CCA). At the network layer, a static routing protocol is used with each node in the network knowing a priori the address of his next hop in order to forward multimedia data. We plan to integrate the proposed video system with a more elaborated routing protocol in which nodes can associate/de-associate with the network and receive/release network addresses such as the one described in [19].

D. Java support

The development of the video functionality onto the sensor nodes required the implementation of the support in the Java environment to allow the interaction between the user and the sensor network and to display both video sequences and still images. The software is based on a simple producer/consumer synchronized multi-thread architecture with two processes that fill in and empty a playout buffer respectively. The receiver thread is directly connected to the sink node through serial communication and it is responsible to arrange the received packets and to reconstruct a frame. If a frame is not received correctly due to the loss of one or more packets, subsequent frames must be discarded until the synchronization is restored. When the receiver thread finds that the sequence of the inner parts of a frame or the frames flow are broken, it stops saving the incoming data and sends a control message to the camera mote.

On the reception of this message, the camera mote blocks the DPCM encoder and proceeds in transmitting a new I-frame. Conversely, the display thread reads the reconstructed frames from the playout buffer, decodes properly I- and P- frames and displays the streamed video using the Java Swing libraries. The display thread is started only when the playout buffer contains a number of frames large enough (5 in our implementation).

Due to the congestion control mechanism, the frame rate is not constant but depends on the actual congestion in the network: hence the display thread adjusts its reading rate based on the estimated number of frames written in the buffer by the receiver thread. This guarantees a fluent video reproduction without consuming the buffer content too fast with respect to the buffer refilling rate. When the playout buffer is empty the display thread stops and waits that the number of frames in the buffer become sufficient.

The graphical user interface shown in Figure 3 allows to explicitly request a video or a still image from the camera motes. Supported resolutions for still images are 640x480 (VGA) and 320x240 (QVGA) with or without JPEG compression. Supported resolutions for video are 320x240, 160x120 and 80x60. In any case, the received frame are rescaled to QVGA with bicubic interpolation for larger video displaying.

IV. Background and System Dimensioning

Before moving to the performance evaluation of the entire video streaming system, it is worth providing here background and dimensioning guidelines to assess on the one hand the actual capacity offered by the reference hardware/firmware platform, and to further discuss how to dimension/optimize the proposed congestion control mechanisms embedded in the video streaming system.

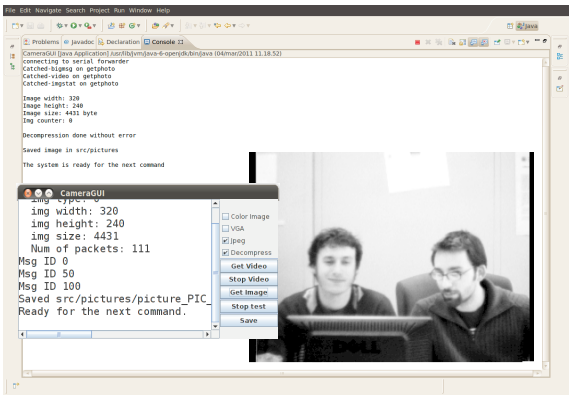


Fig. 3. Graphical User Interface of the video streaming system. The window on the left is the control panel and allows to request the sensor networks for still images or video at different resolutions.

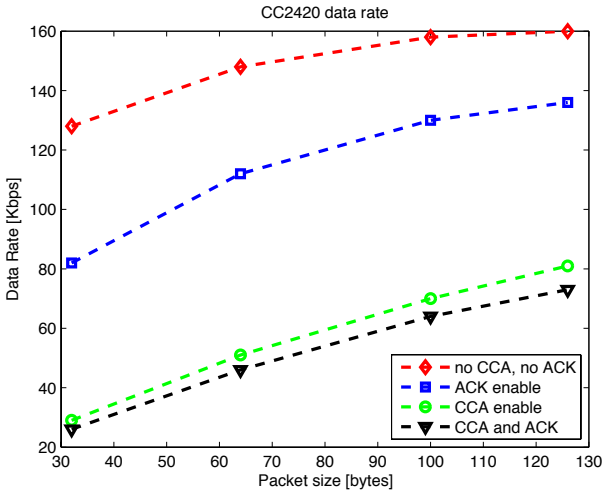


Fig. 4. CC2420 experimental data rate for different configurations of ACK and CCA

A. Benchmarking the Hardware Transmission Capabilities

As explained in Section III-C, the developed video streaming system resorts to the standard IEEE 802.15.4 PHY and MAC layers. To this extent, it is worth evaluating the actual "capacity" offered by the reference hardware/firmware platform. In Figure 4 we plot the experimental data rate achieved on a point to point communication between two nodes with different configuration of the MAC protocol. We sent 10 kilobytes of data segmented in packets of different size (the 802.15.4 maximum packet size is 127 bytes): we tested four configurations in which we switched on/off the clear channel assessment and the acknowledgment of packets. In the best case, when both CCA and ACK were not used, the maximum achievable data rate is about 160 kbps (64% of the nominal 250 kbps data rate of 802.15.4). In our video system, where both CCA and ACK system are used (at least for I-frames), and with a packet size of 56 bytes, the experimental data rate is about 40 - 45 kbps. Increasing the packet size could be a solution to increase data rate, but TinyOS demonstrated problems in correctly receiving packets greater than 50 bytes at high rates. In Section V we report different experimental results that show this behavior.

B. Flow Control Design Guidelines

The proposed congestion control mechanism relies on two thresholds: the *stop threshold* and the *start threshold*. The former represents the maximum buffer occupancy which triggers a stop message sent back to the previous node along the transmission path; the latter determines the time after which the transmitter is reactivated. The proper setting of the threshold values is central for the operation of the whole video delivery mechanism. Roughly speaking, the *stop threshold* should be set high enough to guarantee continuous transmission/relaying of packets while bounding the buffer overflow probability.

In the following, we aim at providing quantitative tools to set the stop threshold. To this extent, we derive close-form expression for the packet dropping probability due to buffer overflow. Let $E[T]$ be the average time taken by the stop message sent by a congestion node to travel back to the traffic source. $E[T]$ obviously depends on the quality of the backward link towards the source. Namely, if such link is characterized by an average Packet Error Rate, p, one can write:

$$E[T] = \sum_{i=1}^{\infty} i(1-p)p^{i-1}(T_{succ} + T_{cca}(i)),$$

being T_{succ} the time to complete a successful transmission on the backward link, which includes the transmission time for the stop message, the transmission time for the acknowledgement, and the propagation delays; similarly, $T_{cca}(i)$ accounts for the time "lost" in the CCA/CSMA procedure at transmission retry i. The packet dropping probability can consequently be estimated as the probability of receiving more than $\Delta = BufferSize - StopThreshold$ packets during $E[T]$, that is,

$$P_{drop} = \sum_{i=\Delta}^{\infty} \frac{\lambda E[T]^i}{i!} e^{-\lambda E[T]}. \quad (1)$$

Eq. 1 describes the dropping probability as a function of the packet error rate, the source rate λ and the parameter Δ. Referring back to Figure 4 and considering a message size of L=18 bytes, an acknowledgement size of L_{ack} =5 bytes and a nominal bit rate in transmission of $c = 120$ kbps, the time for a successful transmission is:

$$T_{success} = L/c + L_{ack}/c.$$

A qualitative estimate for such parameter can be obtained by assuming independency on the transmissions from the source and the transmissions (and retransmissions) of the stop message. Under such assumption, the time for clear channel assessment at the generic transmissions attempt i of the IEEE 802.15.4 standard is equal to 4 backoff periods. Being a backoff period composed of 20 symbols of $16us$ each, the $T_{cca}(i)$=1.28ms.

Figure 5 reports the packet dropping probability as a function of the parameter Δ for different values of the packet error probability in the range [0.1, 0.3] (which are the typical PER values we have experienced in our testbeds) when considering a source rate of 20 kbps. As clear from the figure, a value of parameter Δ around 5 leads to a target packet dropping probability of 10^{-5}.

V. Performance Evaluation

In order to evaluate the performance of the proposed system we carried out several experiments. The reference network architecture is the one showed in Figure 6 where a camera mote acquires a video and sends it to the sink node through all the intermediate nodes. As explained in Section III the routing protocol adopted is static and each node knows a priori the address of its next hop. In the following, each

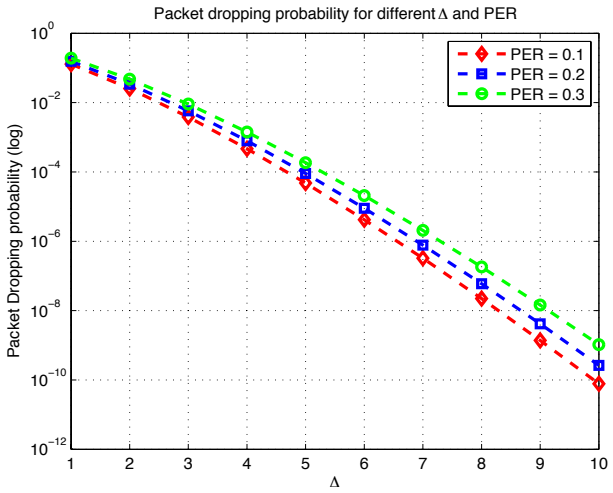


Fig. 5. Packet dropping probability as a function of the Δ parameter; λ =20[kbps].

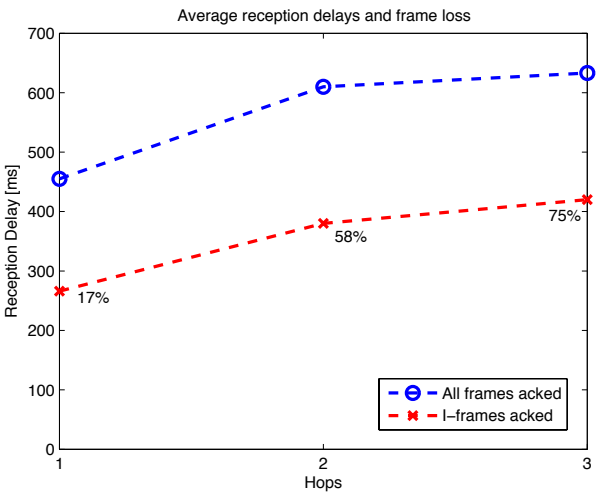


Fig. 7. Average reception delay for two different transport protocol. The blue line refers to the case in which all frames where acked, while the red line corresponds to the selective reliability protocol, in which only I-frame were acked. In this case frame loss percentages are reported for each hops.

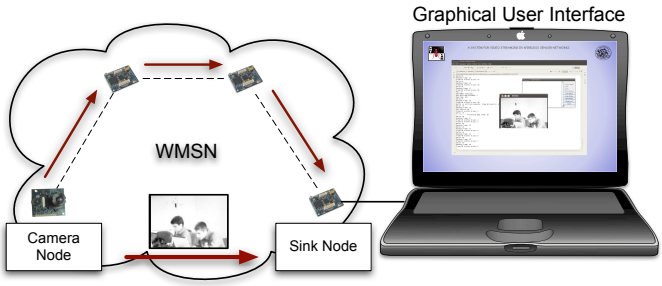


Fig. 6. System architecture

Resolution \ Hops	0	1	2	3
80 x 60	2.034	2.190	2.362	2.939
160 x 120	3.023	3.225	3.808	4.254
320 x 240	4.357	6.580	8.897	9.107

TABLE I
TOTAL SYSTEM DELAYS IN SECONDS, FOR DIFFERENT VIDEO RESOLUTIONS AND HOPS.

test is carried out monitoring our laboratory for five minutes and we averaged the resulting values over the observation window.

In the first experiment we tested the performance of the proposed selective reliability protocol, where each forwarding node requests acknowledgments only for I-frame, while P-frame are allowed to be lost. Figure 7 shows the different performance between the proposed method and a totally reliable protocol that request acks for all frames, in terms of number of lost frames (i.e. *reliability*) and delay (i.e. *timeliness*): when requesting ACKs for all frames, the average reception delay (computed as the time between the reception of the first and the last packet of a frame) varies from 450 ms at 1 hops to 630 ms at 3 hops, but no frames are lost. Conversely when using the selective reliability protocol, the average reception delay is lower (ranging from 260 ms at 1 hops to 420 ms at 3 hops), but the frame loss percentages are higher and higher: for instance, in the 3 hops configuration we experimented that 75% of the frames where lost, resulting in an unacceptable video streaming quality. We thus requested ACKs for any frame in the subsequent experiments. We also evaluated the total system end-to-end delay, including both the reception delay and the delay introduced by the playout buffer at the sink: Table I summarizes the obtained delays in seconds.

In the second experiment we evaluated the performance of the differential hybrid DPCM/DCT encoder in terms of Peak Signal to Noise Ratio (PSNR) and average frame size. Figure 8 shows the average rate-distortion curves for different video resolutions varying the quality factor of the JPEG compression stage. In our implementation the quality factor simply correspond to a scaling on the DCT quantization matrix: higher values mean larger quantization coefficients, resulting in a reduced frame size and visual quality. Since the OV7670 camera chip driver allows to acquire only VGA or QVGA images, in order to obtain resolutions smaller than QVGA we used average downsampling, where each pixel in the downsampled image is obtained by averaging an area of pixels in the original image: this choice does not add excessive computational complexity on the mote's processor but causes worse PSNRs at lower resolutions.

In the last experiment we tested the performance of the video system in terms of frame rate when the video stream is produced from a camera mote respectively at 0, 1, 2 and 3 hops from the sink node. With 0 hops, we mean that the camera node is connected directly through a serial interface to the sink (hence this configuration provides an upper bound to the system performance). Figure 9 reports system frame rates at different resolutions and for different network depths. As expected, the frame rate is limited to very low values, due to the low processing capability of the sensor nodes and limiting channel capacity.

VI. CONCLUSIONS

Multimedia data delivery in wireless sensor networks is a challenging task due to strong limitations imposed by channel capacity and available hardware platforms processing capability. Particularly for video transmission, the research community is very active in finding efficient ways to enable video streaming on multi-hop wireless sensor

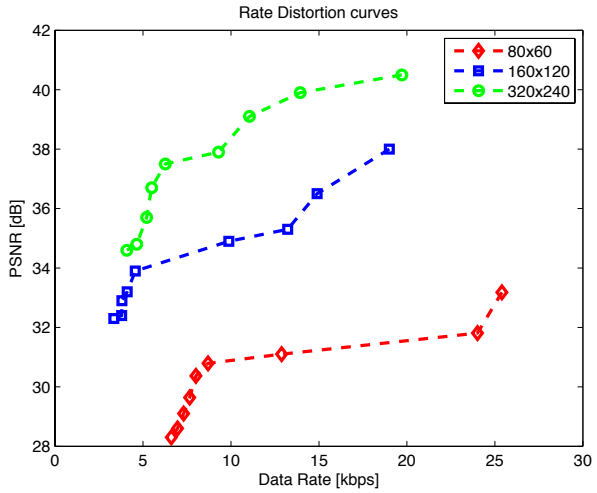


Fig. 8. Experimental rate-distortion curves of the proposed system

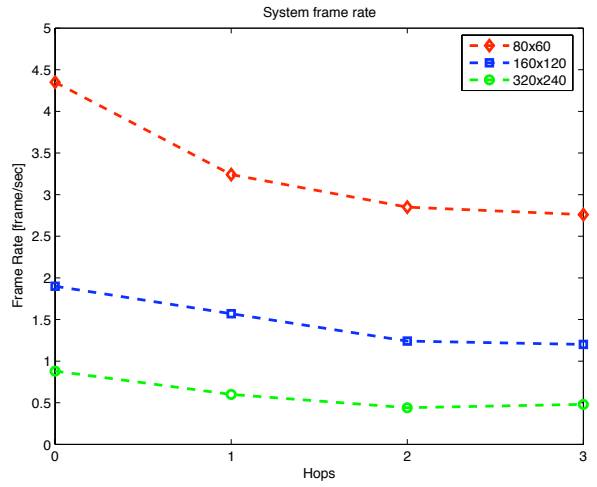


Fig. 9. System frame rate at different hops and for different video resolutions

networks with acceptable Quality of Experience (i.e. visual quality and real-time constraints). Since the most part on the works in this area is based on models or simulations, the main objective of this work was to effectively implement and evaluate a multi-hop streaming video system on commercial hardware. The proposed system is working on top of a network of IntelMote2 sensor nodes, operated by TinyOS. The system features the acquisition of still images and video at different resolutions, and we implemented both a congestion control mechanism and a priority automatic request system to ensure reliability of video streams. Finally, we have carried out several experiments to evaluate system performance in terms of PSNR, frame rate and latency. The complete system (sensor nodes source code and Java engine source code) is made publicly available through a TinyOS contribution, to enable reproducible research. Future work will be focused on the integration of the video system with a more efficient routing protocol, the comparison of the proposed solution with other multimedia encoding techniques together with a detailed study of the energy consumption of both camera, processor and radio modules.

ACKNOWLEDGMENTS

This work has been supported by project LAURA, *Localization and Ubiquitous Monitoring of Patients for Health care support* financed by the Politecnico di Milano.

REFERENCES

[1] A. Redondi, M. Tagliasacchi, M. Cesana, L. Borsani, P. Tarrio, and F. Salice, "LAURA - LocAlization and Ubiquitous monitoRing of pAtients for health care support," in *Advances in Positioning and Location-Enabled Communications, 2010. APLEC '10., IEEE International Workshop on*, 2010.

[2] I. Akyildiz, T. Melodia, and K. Chowdhury, "Wireless multimedia sensor networks: A survey," *Wireless Communications, IEEE*, vol. 14, no. 6, p. 32, dec. 2007.

[3] S. Misra, M. Reisslein, and G. Xue, "A survey of multimedia streaming in wireless sensor networks," *Communications Surveys and Tutorials, IEEE*, vol. 10, no. 4, p. 18, Fourth Quarter 2008.

[4] S. Soro and W. Heinzelman, "A survey of visual sensor networks," *Adv.Multimedia*, p. 21, 2009.

[5] (2011, Mar.). [Online]. Available: http://laura.como.polimi.it/wmsn.html

[6] G. Pekhteryev, Z. Sahinoglu, P. Orlik, and G. Bhatti, "Image transmission over ieee 802.15.4 and zigbee networks," in *Circuits and Systems. ISCAS 2005. IEEE International Symposium on*, 2005, p. 3539.

[7] T. Wark, P. Corke, J. Karlsson, P. Sikka, and P. Valencia, "Real-time image streaming over a low-bandwidth wireless camera network," in *Intelligent Sensors, Sensor Networks and Information. ISSNIP 2007. 3rd International Conference on*, 2007, p. 113.

[8] J. Wang, M. Masilela, and J. Liu, "Supporting video data in wireless sensor networks," in *Multimedia. ISM 2007. Ninth IEEE International Symposium on*, 2007, p. 310.

[9] S. Pudlewski and T. Melodia, "A distortion-minimizing rate controller for wireless multimedia sensor networks," *Computer Communications*, vol. 33, no. 12, pp. 1380 – 1390, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/B6TYP-4YF5R6N-1/2/8fd66b3b83e4529073f41d004b9d7fae

[10] W.-T. Chen, P.-Y. Chen, W.-S. Lee, and C.-F. Huang, "Design and implementation of a real time video surveillance system with wireless sensor networks," in *Vehicular Technology Conference. VTC Spring 2008. IEEE*, 2008, p. 218.

[11] P. Kulkarni, D. Ganesan, P. Shenoy, and Q. Lu, "Senseye: A multi-tier camera sensor network," in *Multimedia, 13th ACM International Conference On,*, 2005.

[12] S. Nath, Y.Ke, P. B. Gibbons, B. Karp, and S. Seshan, "A distributed filtering architecture for multimedia sensors," in *Intel Research Technical Report*, 2004.

[13] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan, "Mesheye: A hybrid-resolution smart camera mote for applications in distributed intelligent surveillance," in *Information Processing in Sensor Networks. IPSN 2007. 6th International Symposium on*, 2007, pp. 360 – 369.

[14] I. Politis, M. Tsagkaropoulos, and S. Kotsopoulos, "Optimizing video transmission over wireless multimedia sensor networks," in *Global Telecommunications Conference. IEEE GLOBECOM 2008. IEEE*, 2008, p. 1.

[15] S. Guo and T. Little, "Qos-enabled video streaming in wireless sensor networks," in *INetwork Computing and Applications (NCA), 2010 9th IEEE International Symposium on*, 2010, p. 214.

[16] S. Qaisar and H. Radha, "Multipath multi-stream distributed reliable video delivery in wireless sensor networks," in *Information Sciences and Systems, (CISS 2009). 43rd Annual Conference on*, 2009, pp. 207 – 212.

[17] S. Gonzalez-Valenzuela, H. Cao, and V. Leung, "A multi-channel approach for video forwarding in wireless sensor networks," in *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, 2010, p. 1.

[18] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proceedings of the IEEE*, vol. 93, no. 1, p. 71, jan. 2005.

[19] L. Borsani, A. Redondi, S. Guglielmi, and M. Cesana, "Tree-based routing protocol for mobile wireless sensor networks," in *Proceedings of the 8th IEEE International Conference on Wireless On-Demand Network Systems and Services (WONS2011)*, 2011, pp. 158–163.