J Supercomput
DOI 10.1007/s11227-010-0475-2

Enabling low bit-rate and reliable video surveillance over practical wireless sensor network

Min Chen · Sergio González · Huasong Cao · Yan Zhang · Son T. Vuong

Abstract With the rapid development of hardware and embedded systems, wireless sensor networks (WSNs) are being developed for surveillance applications. While meriting more in-depth research and development, deploying a practical WSN for surveillance is challenging due to the limited power and bandwidth of the battery-operated sensor nodes. In this paper, we first propose an energy-efficient image transportation strategy through motion detection. In case of data delivery over long distance, this paper further investigates the use of cooperative communications to design a reliable image transmission scheme over WSN, and demonstrates its effectiveness in improving network reliability in wireless multimedia sensor networks.

M. Chen (✉)
School of Comp. Sci. & Eng., Seoul National University, Seoul, South Korea
e-mail: minchen@ieee.org

S. González · H. Cao
Dept. of Elect. & Comp. Eng., Univ. of British Columbia, Vancouver, Canada

S. González
e-mail: sergiog@ece.ubc.ca

H. Cao
e-mail: huasongc@ece.ubc.ca

Y. Zhang
Simula Research Laboratory, Lysaker, Norway
e-mail: yanzhang@ieee.org

Y. Zhang
IFI, University of Oslo, Oslo, Norway

S.T. Vuong
Dept. of Computer Science, Univ. of British Columbia, Vancouver, Canada
e-mail: vuong@cs.ubc.ca

Published online: 09 September 2010

1 Introduction

Wireless Sensor Networks (WSNs) consist of a group of geographically distributed, low-power sensors, which monitor physical or environmental conditions such as temperature, humility, pressure, sound and vibration [1]. The collected data is then sent to a central node using wireless transmission. Investigation of WSNs has become significant in recent years because of their applications in everyday life for gathering information in their deployment setting. While traditional sensor applications have been widely studied, the use of WSN for video surveillance is more challenging and merits more in-depth research and development [2–6]. In this paper, we focus particularly on image transfer over a WSN.

As technology evolves to provide more possibilities in computer and real world interactions, significant developments in the WSN field have been recently introduced. In this paper, we consider a simple WSN that only contains three IMote2 nodes [7] and transmits image data in JPEG format. The IMote2 sensor node from Crossbow Technology features an Intel PXA271 XScale CPU with its clock frequency up to 416 MHz, and an embedded IEEE 802.15.4 compatible radio chip (TI CC2420) with on-board antenna. Though IMote2 is capable of performing limited in-node image-processing, there is a limitation on the size of the data packet being transmitted in accordance to IEEE 802.15.4 protocol. Thus, it is impossible for an image to be sent from one node to another without segmenting it into multiple fragments of smaller size.

As a consequence of the above-mentioned problem, implementing an efficient method for segmentation and reassembly are key to successfully and reliably transmitting image segments over WSN links. Otherwise, the system may become power-inefficient and prone to data loss, which are detrimental to the overall system's performance. Consequently, a data buffer algorithm is used to maintain a collection of all packets, and a cooperative node is added into the network to provide a backup source for the data. According to our experimental results, the buffer algorithm guarantees a successful and complete image transfer. The addition of the cooperative node into the network significantly increases the reliability and transmission distance. These design considerations are essential for the efficiency of the performance, and for lowering the cost of a WSN.

The contributions of this paper are summarized as follows:

- We advance a complete system design that can be readily employed to provide effective surveillance of valuable assets.
- We discuss our practical experiences after implementing a prototype system in commercially available devices, contrary to simply running computer simulations.
- We introduce a simple but effective protocol to recover image data packets that are lost or corrupted by diverse problems occurring in the wireless medium.
- We report the performance results of our prototype, which show the efficiency of the wireless surveillance system, and can serve as a useful reference for future improvements.

The paper is organized as follows. In Sect. 2, we discuss the architecture of the proposed wireless battery-operated low bit-rate point-to-point video sensor system.

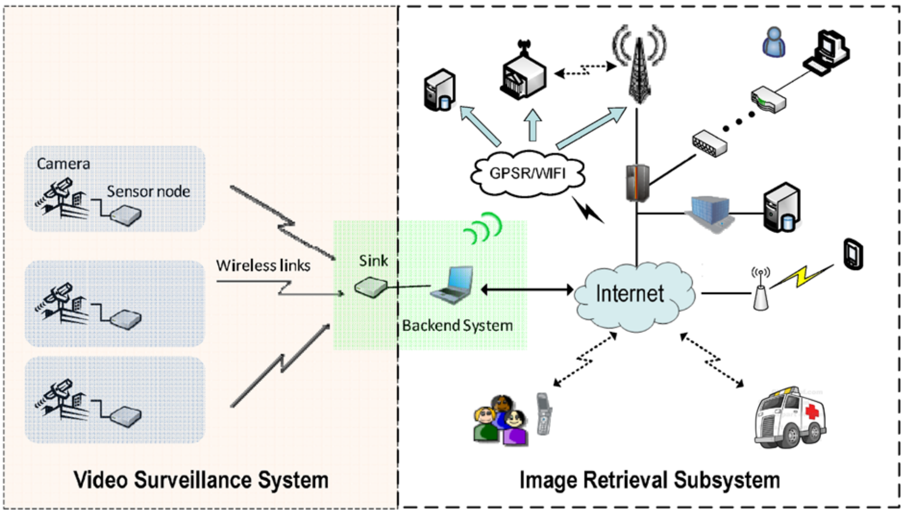

Fig. 1 Illustration of system architecture

We explain the detailed system operations in Sect. 3. In Sect. 4, a cooperative transmission scheme for reliable image transfer over the proposed video sensor system is described. Section 5 shows the results of the experiments, and we conclude our paper in Sect. 6.

2 System architecture

As shown in Fig. 1, the architecture of the video sensor system is comprised of two main subsystems, i.e., a video surveillance network and an image retrieval system.

2.1 Video surveillance system

The video surveillance system is formed by wireless sensor nodes and low-power, miniature video cameras attached to them that individually capture and preprocess images in the deployment area. The sensed images are forwarded to data sink, which is usually a computer that controls the operation of the sensor nodes and forwards the snapped images by means of a communications device that finally relays images to the image retrieval subsystem. In practice, we use the following devices to construct the video surveillance system:

- *IMote2 sensor node*: The IMote2 sensor node from Crossbow Technology features an Intel PXA271 XScale CPU with its clock frequency up to 416 MHz, and an embedded IEEE 802.15.4 compatible radio chip (TI CC2420) with on-board antenna. This radio chip can be tuned to one of the 11 channels numbered from 11 (@ 2.405 GHz) to 26 (@ 2.480 GHz), each separated by 5 MHz according to IEEE 802.15.4 specifications. RF transmission power is programmable from 0 dBm (1 mW) to −25 dBm. Note the power has direct effect on the range of communications between two radio chips.

- *Image capturing device*: A COMedia Ltd.'s C328R [8, 9] VGA camera module with JPEG engine is integrated with each IMote2 sensor node, which provides the image sensor in our platform. Image resolutions supported are 80×60, 160×128, 320×240, and 640×480 pixels. It can only be interfaced through a serial port, and no flow control is needed for the serial communications. Compared to a USB-interfaced camera module, the C328R provides the widely available Universal asynchronous receiver/transmitter (UART) port, and consumes less power but supports a lower data rate due to the limitation of the serial port (the highest data rate is 115,200 bps). Upon receiving commands from the IMote2, it responds either with an acknowledgment packet, or with image data packets. Turning on and off the camera is controlled by the commands from IMote2 sensor node as well.
- *Data sink*: This element is implemented by another IMote2 board as those used for connecting cameras, but with different network connections and software implementations. The data sink also works as the gateway, interpreting commands from Image Retrieval System, coordinating multiple IMote2 sensor nodes with cameras, and relaying image data from those nodes to the data server.
- *External communications device* is used for interfacing with the data server in the Image Retrieval System, and it can be an Ethernet/WLAN Network Interface Card (NIC) with an OEM board or a USB dongle. In our current implementation, the image data are dumped from the IMote2 sensor board to a desktop/laptop through a USB link.

2.2 Image retrieval system

The image retrieval system is a customized data server that stores and processes the images and other information sent by the gateway subsystem through a public communications network such as: (a) the cellular backhaul, or (b) the Internet. It also interfaces communications between the client and the video surveillance subsystem through the same network access systems.

The data server runs a FTP and a Web server. The data server is implemented by a commercially available laptop configured to service FTP and HTTP requests. In this regard, the surveillance system connects to the data server through an FTP connection to store the latest images. These images are stored in the server's hard drive for an unspecified amount of time. At the same time, the Web server has access to the same storage space where these images are being stored, and can be readily retrieved as requested by users. For a real setting, implementation costs for the data server subsystem depend on the software technology employed. In this regard, the most popular platforms are the commercial, license-based Microsoft server technologies, and the open-source Linux OS. Whereas the former technology is widely employed, it also incurs significantly more setup and maintenance overhead costs. On the other hand, Linux has shown to provide an excellent alternative to Microsoft technology. Linux-based server platforms are also widely used, and their implementation costs can be greatly reduced due to their being license-free.

The architecture of the image server subsystem is shown in Fig. 2. The front-end of the data server and the entire system is a Web-based interface that allows the user to control the monitoring system from any location where an Internet connection is

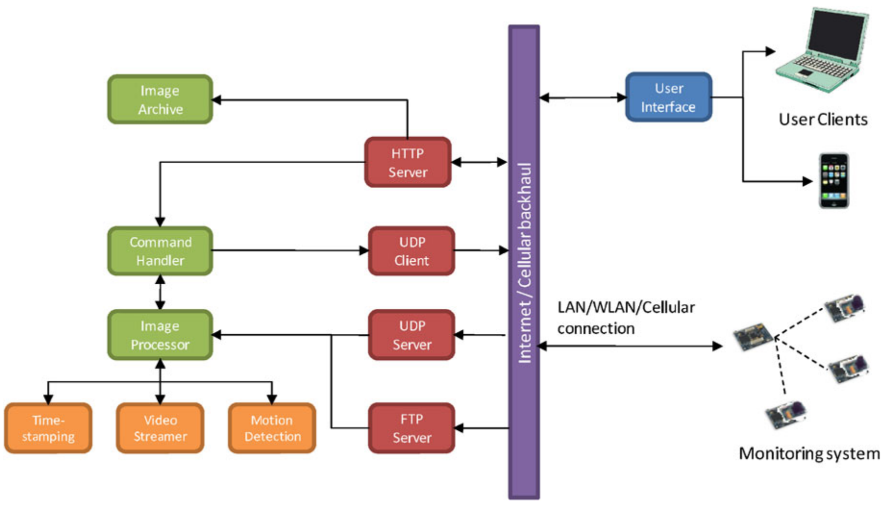

Fig. 2 The architecture of image retrieval system

available. Any command that the user issues through this interface is first processed by a command handler module, whose access is in turn controlled by an HTTP (Web) server. The command is finally forwarded to the monitoring subsystem by means of a separate client software module. At the gateway, device in the monitoring subsystem processes this request accordingly, and may reply with the corresponding information through a UDP server and a FTP server. A separate image processor module is in charge of modifying and archiving images received directly from the monitoring system for immediate retrieval by the user if so desired.

3 System operation

There are four modes for an IMote2 sensor node equipped with a camera. We use "camera mode" here to refer to possible operational states of such a node:

- *ON*: The camera is turned on.
- *OFF*: The camera is turned off.
- *INIT*: The camera will: capture one image, store it as the "background," and upload it to the sink. Subsequent captured images will be compared to this background image for detecting motion.
- *LIVE*: The camera works in live mode, where the camera will upload images without performing motion detection.

According to the different functionalities of a packet, we categorize the packets into four categories: mode control, image request, image data, and message. The description of each packet is shown as follows:

- *Mode control*: (1) *Cam(ON)*: Turn on camera; (2) *Cam(OFF)*: Turn off camera.
- *Image request*: (1) *ImgReq(INIT)*: Request an initial image from the camera; (2) *ImgReq(LIVE)*: Request the camera to capture an image in LIVE mode.

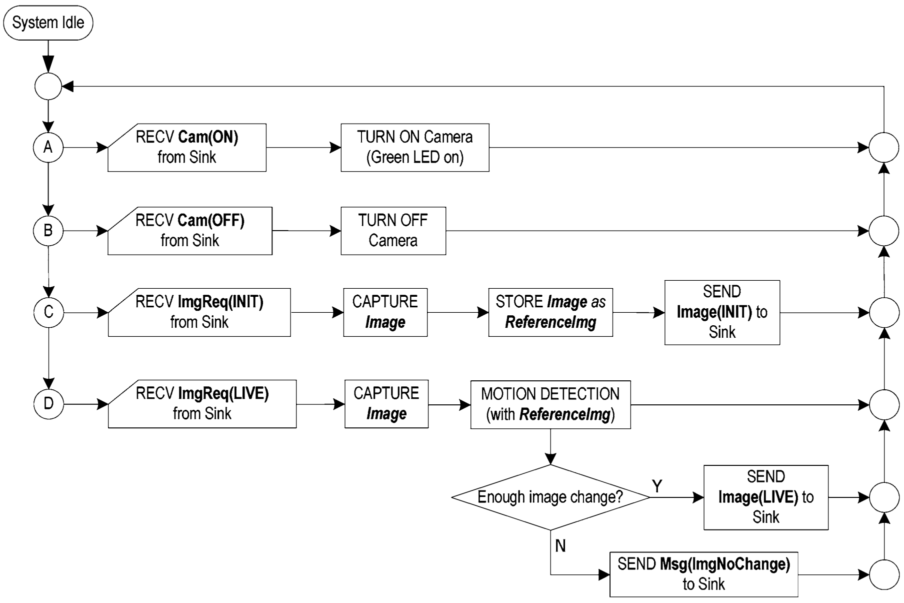
System Idle
A
RECV Cam(ON) from Sink
TURN ON Camera (Green LED on)
B
RECV Cam(OFF) from Sink
TURN OFF Camera
C
RECV ImgReq(INIT) from Sink
CAPTURE Image
STORE Image as ReferenceImg
SEND Image(INIT) to Sink
D
RECV ImgReq(LIVE) from Sink
CAPTURE Image
MOTION DETECTION (with ReferenceImg)
Enough image change?
Y
SEND Image(LIVE) to Sink
N
SEND Msg(ImgNoChange) to Sink

Fig. 3 Flow chart of the operations at camera sensor node

- *Image data*: (1) *Image(INIT)* denotes the image first captured by the camera since it is turned on; (2) *Image(LIVE)* denotes the image captured by the camera in LIVE mode.
- *Message*: *Msg(ImgNoChange)*: When the camera does not detect motion, a simple message indicating no image change will be delivered instead of sending the whole image data.

When a camera node is in LIVE mode, the system compares the latest captured image with the reference (background) image. This comparison is made by employing the MSE (Mean Square Error). If the calculated MSE is larger than a preset threshold, we consider that an image anomaly has been detected, and the sink is immediately notified. The camera keeps operating in LIVE mode capturing images, until no anomalies with respect to the reference image are further detected. Note that a reference image is taken every time a camera node is switched on in order to guarantee that it is an up-to-date. The flow chart of the operations at the camera sensor node is shown in Fig. 3. The alphabet index of each procedure in Fig. 3 is identical to the one illustrated as follows:

A. When the camera-IMote2 node receives Cam(ON) packet from sink, it turns on its camera.
B. When the camera-IMote2 node receives Cam(OFF) packet sent by the sink, it turns off its camera.
C. When the camera-IMote2 node receives ImgReq(INIT), it captures the latest image, and stores it as its reference image. The reference image is used to perform motion detection later. Then, the image is sent to the sink in packet format of Image(INIT).

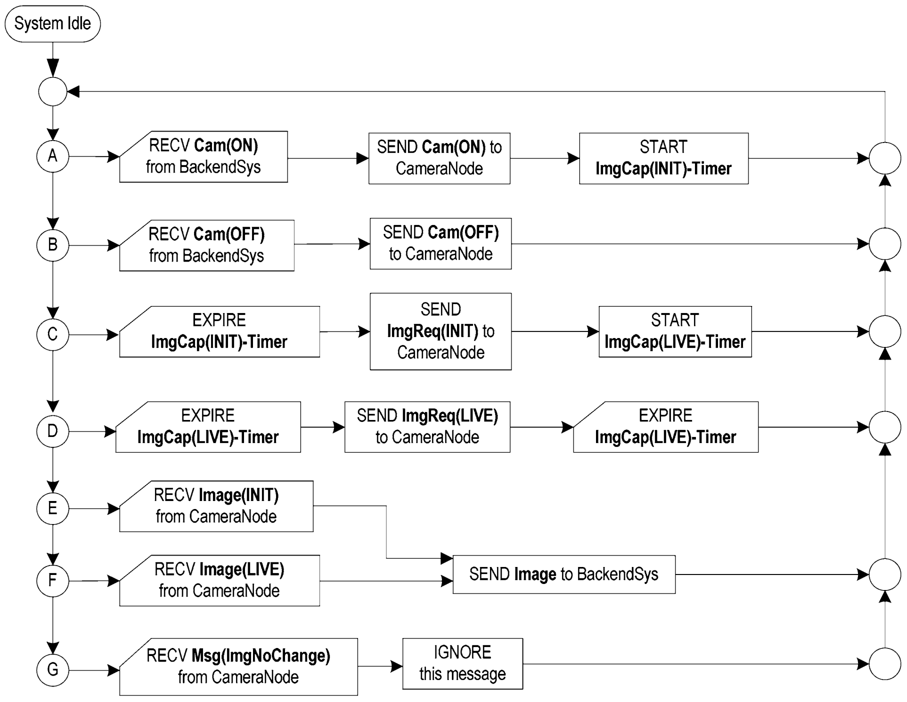
System Idle
A
RECV Cam(ON) from BackendSys
SEND Cam(ON) to CameraNode
START ImgCap(INIT)-Timer
B
RECV Cam(OFF) from BackendSys
SEND Cam(OFF) to CameraNode
C
EXPIRE ImgCap(INIT)-Timer
SEND ImgReq(INIT) to CameraNode
START ImgCap(LIVE)-Timer
D
EXPIRE ImgCap(LIVE)-Timer
SEND ImgReq(LIVE) to CameraNode
EXPIRE ImgCap(LIVE)-Timer
E
RECV Image(INIT) from CameraNode
F
RECV Image(LIVE) from CameraNode
SEND Image to BackendSys
G
RECV Msg(ImgNoChange) from CameraNode
IGNORE this message

Fig. 4 Flow chart of the operations at sink node

D. When the camera-IMote2 node receives ImgReq(LIVE) from the sink, it captures the latest image. Then motion detection is performed between the reference image and the latest one. If significant portion of the latest image has changed with respect to the reference image, then the latest camera shot is forwarded to the sink. Otherwise, a message will be sent to the sink indicating that there is no image change.

The flow chart of the operations at data sink is shown in Fig. 4. The alphabet index of each procedure in Fig. 4 is identical to the one illustrated as follows:

A. When the sink receives Cam(ON) from the back-end system, it sends the packet to camera node, and starts ImgCap(INIT)-Timer.
B. When the sink receives Cam(OFF) from the back-end system, it sends the packet to camera node.
C. When ImgCap(INIT)-Timer expires, the sink sends ImgReq(INIT) to camera node, and starts ImgReq(LIVE)-Timer, which means the system will work at LIVE mode in the following time.
D. When ImgReq(LIVE)-Timer expires, the sink sends ImgReq(LIVE) to the camera node, and starts a new ImgReq(LIVE)-Timer. This will create a loop to continue the video surveillance process until receiving a Cam(OFF) command from the backend system.
E. When the sink receives an Image(INIT) command from the camera node, it uploads the image to the back-end system.

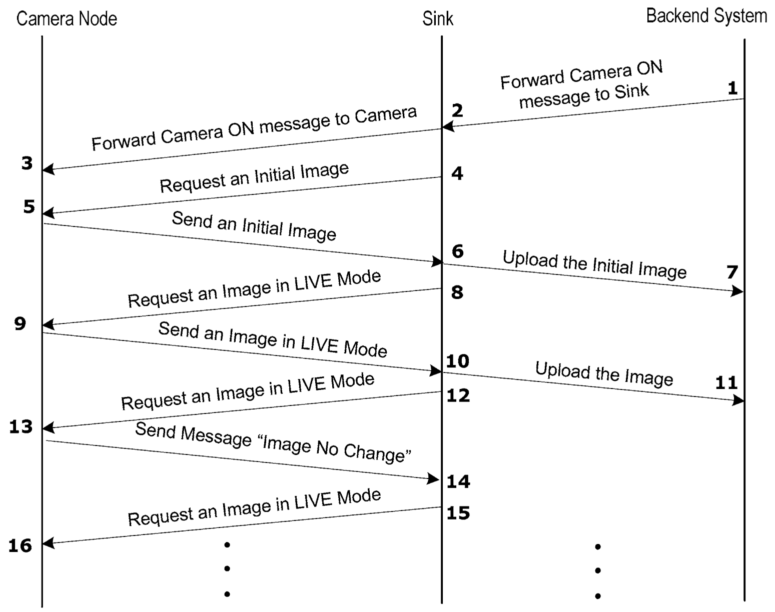
Camera Node
Sink
Backend System
Forward Camera ON message to Sink
1
2
Forward Camera ON message to Camera
3
Request an Initial Image
4
5
Send an Initial Image
6
Upload the Initial Image
7
Request an Image in LIVE Mode
8
9
Send an Image in LIVE Mode
10
Upload the Image
11
Request an Image in LIVE Mode
12
13
Send Message "Image No Change"
14
Request an Image in LIVE Mode
15
16

Fig. 5 A typical example to illustrate the system inter-operation

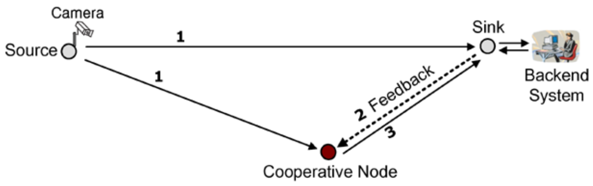

Fig. 6 Illustration of cooperative scheme for reliable image delivery over wireless sensor networks

F. When the sink receives Image(LIVE) command from the camera node, it uploads the image to the back-end system.

G. When the sink receives a Msg(ImgNoChange) command from the camera node, no action is taken: the message is simply ignored.

Figure 5 depicts a typical procedure which contains most of the important inter-operations among the key components in our system. We use numerical symbols (e.g., 1, 2, 3) to denote the sequence number of an event, and the order of the event SeqNum progressively increases with time.

4 Cooperative communications for reliable image delivery over WSNs

4.1 Architecture

As shown in Fig. 6, the camera sensor node functions as the source node, which captures a picture of the environment, and then segments the image into a small packet with an assigned sequence number. It then transmits the image data packet one by

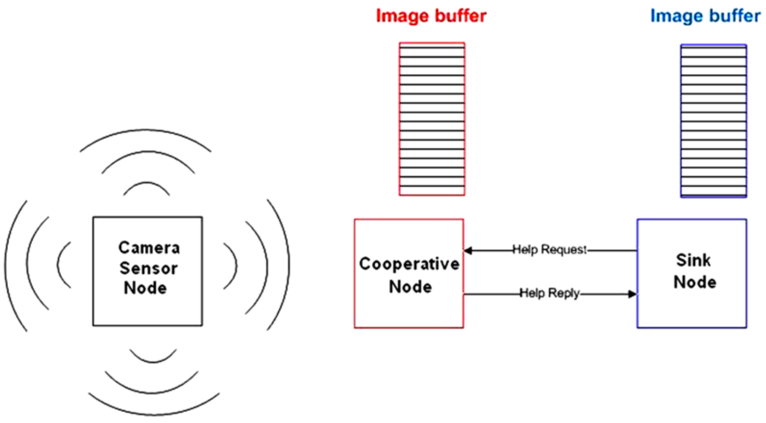

Fig. 7 Illustration of the functionality of the cooperative node

one. At the same time, the sink node tries to collect all the data packets, before temporarily storing and reassembling them according to the their segment number to recreate the original image. This process executes continuously until the back-end system interrupts the sink node. However, during the packet transmission, it is always possible that some packets will get corrupted before being received by the gateway node due to the distance, interference, or data overflow. In this situation, the rebuilt image might result in a distorted image. To provide a more reliable transfer protocol, we introduce a cooperative node in-between the source and sink nodes. When the camera sensor node transmits the image packets, both the cooperative node and the sink node will maintain an image buffer using the on-board memory.

As illustrated in Fig. 7, the cooperative node is placed close to the camera node to guarantee that no packet is lost due to long distance, and will have a copy of the complete image with all the data packets intact. On the other hand, the sink node is placed at a further distance, and due to interference and transceiver range, packet loss is still possible as mentioned above. To solve this problem, the sink node requests the missing packet to the cooperative node using the corresponding sequence number. The cooperative node then forwards the requested packets to the gateway node. This reduces the probability of reassembling a distorted image. Finally, the camera node deletes the image immediately after transmitting the missing packet in order to free up memory.

4.2 Cooperative mechanism

4.2.1 Request message

After sorting the packets in the image buffer by their sequence numbers, the sink node can identify the lost packets. The request message will then be generated by the sink node to indicate the list of lost packets. The packet is then sent to the cooperative node and the sink node waits for the response. On the reception of the request message, the cooperative node replies with a series of data packets which are indicated in the checklist of the lost packets. Once all the requested packets were successfully resent, the sink node is able to rebuild the image. Otherwise, next requested packet is sent.

4.2.2 Operations of the cooperative node

The cooperative node overhears all the packet transmissions. Similarly to the sink node, the cooperative node also maintains an image buffer to store the incoming data to satisfy the sink node in case of a packet request. The cooperative node overhears the image packets and image header packets sent by the camera node, as well as the request message from the sink node. Upon receiving a packet, the cooperative node first identifies the type of packet and does the corresponding operation.

The image buffer of the cooperative node is initialized as an empty buffer, and the data will be stored in sequential order when a packet is received. If the received packet is an image header packet, it means that the previous image has been successfully received and built by the sink node; hence, there is no need to keep the old image data. Therefore, the image buffer will be flushed and re-initialized as an empty buffer. The node will continue to ping for the next data packet that is being transmitted in the network.

If the received packet is a request message sent from the sink node, then the cooperative node first iterates through its own image buffer to check whether it actually maintains a copy of the corresponding packet. If the requested packet is present in the buffer, the cooperative node will marshal the data and send it to the sink node.

If the requested packet cannot be found in the image buffer, it indicates that the cooperative node did not overhear that particular packet and therefore it could not satisfy the sink node's request. As a result, the sink node would not be able to build the complete image due to the missing packets. However, the probability of this situation is relatively low as the cooperative node is strategically placed within a reliable transmission distance of the camera sensor node. The flow chart of the operations at the cooperative node is shown in Fig. 8.

5 Experiment and results

In this section, we will demonstrate the ability to capture images from battery-operated sensor nodes by means of low data-rate point-to-point communications to display them on a remote laptop through the image retrieving subsystem. The video frame-rate yields approximately 0.5 fps for the resolution of 320 × 240 pixels tested in the live demo. These are JPEG color images that occupy just over 4 KB of data.

To effectively evaluate our implementation, we conducted several tests to collect the performance-related data. The result of our experiments will be described as follows.

5.1 Image encoding and transmission test

We tested our image encoding and transmitting protocol by performing basic send/receive image operations using two IMote2, the objective of performing this

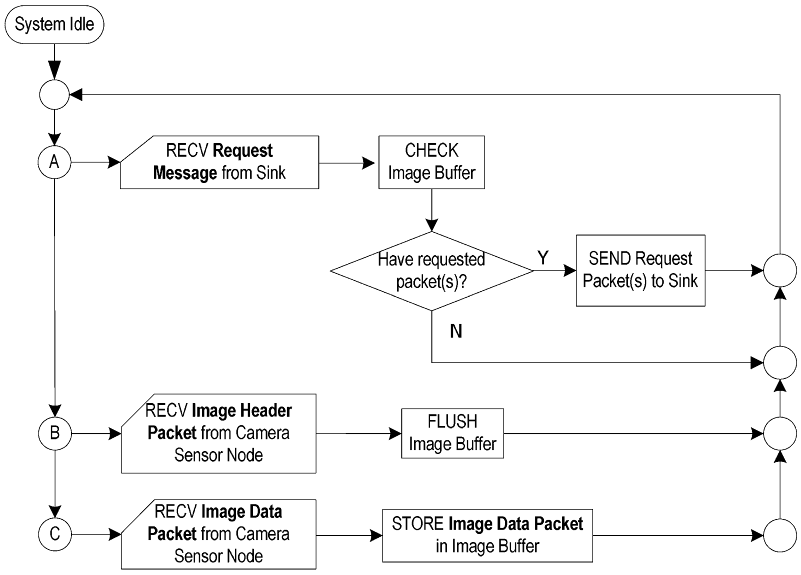
System Idle
A
RECV Request Message from Sink
CHECK Image Buffer
Have requested packet(s)?
Y
N
SEND Request Packet(s) to Sink
B
RECV Image Header Packet from Camera Sensor Node
FLUSH Image Buffer
C
RECV Image Data Packet from Camera Sensor Node
STORE Image Data Packet in Image Buffer

Fig. 8 Flow chart of the operations at cooperative node

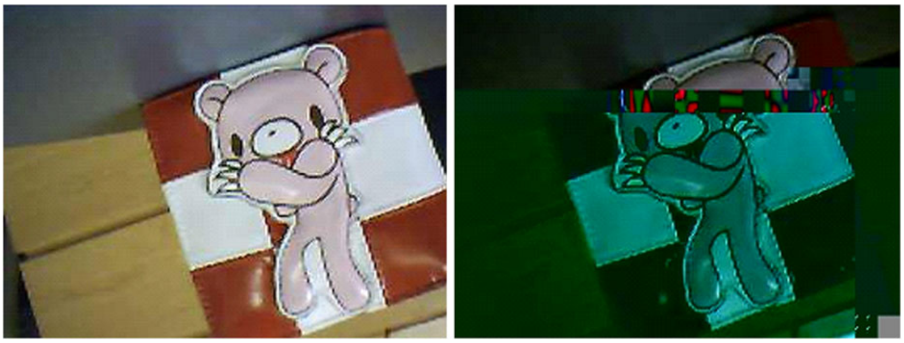

Fig. 9 Distorted image due to loss/corrupted packet

experiment being to make sure the image would be segmented and rebuilt correctly on the nodes. We used an on-board image to carry out the test and compared the pre-stored image on the sender node with the rebuilt image on the receiver node to evaluate the implementation.

Figure 9 illustrates an unsuccessful image transmission where image on the right is heavily distorted.

5.2 Real-time image transmission test

In this section, we equipped one of the IMote2 with the C328R camera and triggered it to capture a real-time image and forward it to the receiver node. The experiment was carried out the same way as in the previous test, where we compared the original

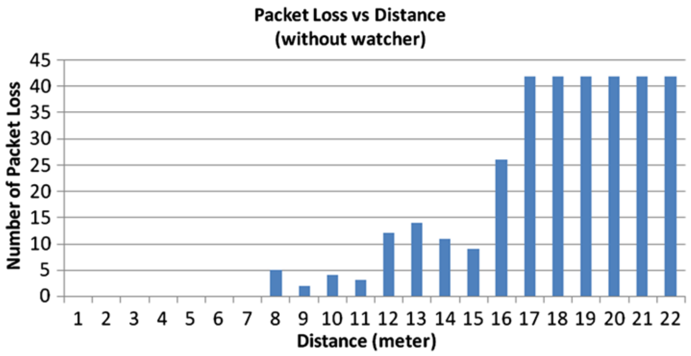

Fig. 10 The impact of transmission distance on the number of packet loss (without cooperative node)

and the rebuilt images. A successful transmission would have no visual difference between the original and the rebuilt images. After ensuring that the real-time image was successfully sent successfully, we moved on to the next phase, which is to add in the cooperative node. The test plan and experiment result are described in the next section.

5.3 Reliable distance test

In order to configure the reliable transmission distance for placement of the watcher node, first we performed the basic image transfer test repeatedly using two IMote2 nodes. We added a counter on the sink node function to keep track of the number of lost packets during each transmission. Then we increased the distance between the two nodes and performed the same test again. The packet loss rate is illustrated in Fig. 10.

As shown in the figure, starting at a distance 8 meters apart, the image packet began being dropped. At the distance of 17 meters, all packets were dropped and the sink node was unable to communicate with the sender (camera) node for packet transmission. Based on this test result, we found the maximum distance that guarantees reliable data transmission was 7 meters.

Next, we tested the network transfer with the presence of the cooperative node. In order to maximize the reliable transfer distance between the sink and the camera node, we placed the cooperative node 7 meters apart from the camera node. As mentioned previously, the cooperative node overhears all the packet transmissions and keeps a copy of the data packets being transmitted; after receiving a request message from the sink node, the cooperative node marshals and forwards the requested data to the sink node. Again, a counter is maintained on the sink node to keep track of the number of lost packets. However, if the sink node receives a lost packet from the cooperative node, it then decrements the counter. A count of zero indicates that there is no packet loss, and that the image was built successfully. Figure 11 indicates that, with the help of a cooperative node, the image packets were guaranteed to transmit successfully up to a distance of 10 meters between the sink and camera node.

In order to illustrate the effectiveness of using a cooperative node, Fig. 12 shows a percentage comparison of lost packets between a WSN with and without a coopera-

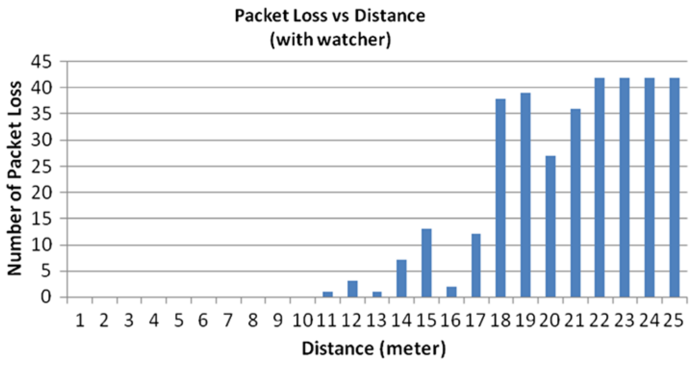

Fig. 11 The impact of transmission distance on the number of packet loss (with cooperative node)

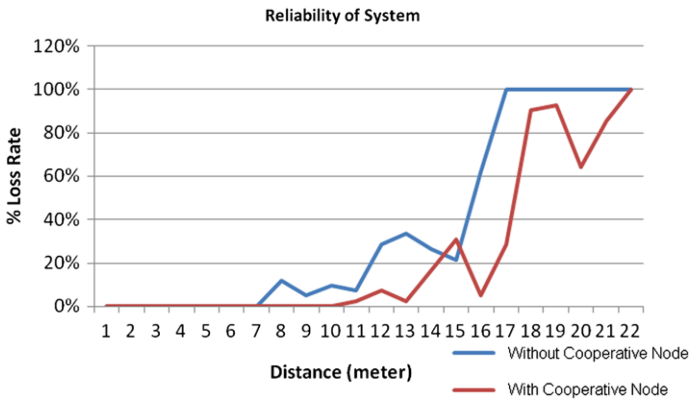

Fig. 12 The impact of transmission distance on the reliability of the video sensor system

tive node. It is clear that the cooperative node helps increasing the maximum reliable distance for packet transmission of a WSN.

We have also compared the original sent image on the camera node with the one stored and received on the cooperative node and the sink nodes. Figure 13 indicates a transmission with four lost packets retrieved from the cooperative node. It shows that the two images are perfectly the same, meaning that the cooperative node has responded to the help request and successfully resent data to the sink node.

6 Conclusion

This paper investigated the use of WSN for video monitoring. We first propose an energy-efficient image transportation strategy through motion detection. Then, we design an efficient image encoding method and implement a reliable transmission protocol for this small WSN, and also to evaluate its corresponding performance. Our experiments and test results have shown that our data recovery protocol is successful. Also, the cooperative node handles the help request correctly. Our implementation

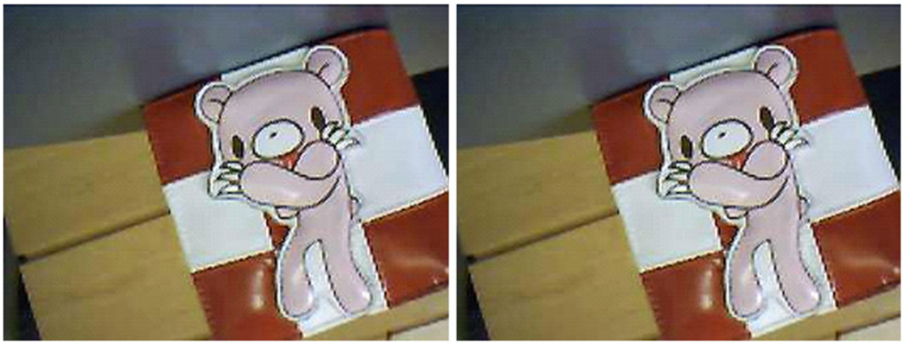

Fig. 13 Successful data packet retrieving from the cooperative node

has improved the maximum reliable distance and the overall performance for a small WSN. However, further development on the starting/ending of image data sequence is required to prevent unnecessary dependency.

References

1. Al-Karaki J, Kamal E (2004) Routing techniques in wireless sensor networks: a survey. IEEE Pers Commun 11(6):6–28
2. Chen M, Leung VCM, Mao S, Yuan Y (2009) Directional controlled fusion in wireless sensor networks. Mob Netw Appl 14(2):220–229
3. Teixeria T, Lymberopoulos D, Culurciello E, Aloimonos Y, Savvides A (2006) A lightweight camera sensor network operating on symbolic information. ACM SenSys, workshop on distributed smart cameras
4. Rowe A, Goel D, Rajkumar R (2007) FireFly mosaic: a vision-enabled wireless sensor networking system. In: Proc. IEEE RTSS'07
5. Chen M, Mao S, Yuan Y, Leung V (2008) Video communications over wireless sensor networks Broadband mobile multimedia: techniques and applications. CRC Press, New York. ISBN-10: 1420051849; ISBN-13: 978-1-4200-5184-1
6. Chen M, Leung V, Shu L, Chao H-C On multipath balancing and expanding for wireless multimedia sensor networks. Int J AD Hoc Ubiq Comput (to appear)
7. Crossbow Ltd., IMote2. http://www.xbow.com/Products/productdetails.aspx?sid=253
8. Stanford, C328R user manual. http://ssdl.stanford.edu/ssdl/images/stories/AA236/0708A/Lab/Rover/Parts/cam_c328.pdf
9. COMedia Ltd., C328r JPEG Camera Module with UART Interface, http://www.comedia.com.hk/fp2008/Spec_PDF/C328R.pdf