Wireless Line Sensor Network for Distributed Visual Surveillance

Mangesh Chitnis
Scuola Superiore Sant'Anna, Italy
Indiana Univ. Purdue Univ., USA
m.chitnis@sssup.it

Yao Liang, Jiang Yu Zheng
Indiana University-Purdue University
Indianapolis, IN, USA
{yliang, jzheng}@cs.iupui.edu

Paolo Pagano, Giuseppe Lipari
Scuola Superiore Sant'Anna
Pisa, Italy
{p.pagano,g.lipari}@sssup.it

ABSTRACT

Wireless sensor networks (WSNs) play a crucial role in visual surveillance for automatic object detection, such as real-time traffic monitoring, vehicle parking control, intrusion detection, and so on. These online surveillance applications require efficient computation and distribution of complex image data over the wireless camera network with high reliability and detection rate in real time. Traditionally, such applications make use of camera modules capturing a flow of two dimensional images through time. The resulting huge amount of image data impose severe requirements on the resource constrained WSN nodes which need to store, process and deliver the image data or results within a certain deadline. In this paper we present a WSN framework based on line sensor architecture capable of capturing a continuous stream of temporal *one dimensional image* (*line* image). The associated one dimensional image processing algorithms are able to achieve significantly faster processing results with much less storage and bandwidth requirement while conserving the node energy. Moreover, the different operating modes offered by the proposed WSN framework provide the end user with different tradeoff in terms of node computation versus communication bandwidth efficiency. Our framework is illustrated through a testbed using IEEE 802.15.4 communication stack and a real-time operating system along with one dimensional image processing. The proposed line sensor based WSN architecture can also be a desirable solution to broader multimedia based WSN systems.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication networks**]: Distributed Systems-Distributed Applications; C.3 [**Special-Purpose and Application-Based Systems**]: Real-time and Embedded Systems; D.2.11 [**Software Engineering**]: Software Architectures-Domain Specific Architectures

General Terms

Algorithms, Design, Experimentation.

Keywords

Wireless Sensor Network, Line Sensor, IEEE 802.15.4, Real time Operating System, Multimedia, Image Processing

1. INTRODUCTION

Distributed visual surveillance based on wireless sensor network is of paramount importance. Its notable applications include traffic monitoring, vehicle parking control, intrusion detection, and so on, which requires to identify, recognize and classify the objects in order to take appropriate action. These applications need to continuously capture images in order to monitor certain events. The goal of such applications is to capture images as fast as possible, process these images with minimum amount of computation and transfer the image information or image itself in a bandwidth-limited distributed system to monitor and identify the events with high reliability in real time.

In order to satisfy these requirements for visual surveillance based on WSN, we present a novel wireless line sensor network (WLSN) architecture in this paper. A line senor generates a stream of *one-dimensional* images instead of a stream of the traditional *two-dimensional* images [13]. That is, the line sensor continuously captures data on a single line in the view. In effect, this is similar to reading a single row or column of a two-dimensional picture frame of a continuous movie stream, The process of which produces a stream of temporal one-dimensional image vectors separated by the frame sampling rate. However we need to achieve a similar goal not for a stationary object but for moving objects. In this case it is enough to keep the line sensor stationary to capture the moving object. As shall be shown in this paper, the use of line images generated by line sensor over the traditional two-dimensional images has dramatic advantages in bandwidth, memory, CPU speed and power constrained WSN applications.

The rest of paper is organized as follows. Section 2 describes the state of art in the field of wireless multimedia sensor networks (WMSN). In Section 3, we present our WLSN hardware/software architecture. In Section 4, we devise a suit of algorithms for line image processing. In Section 5, we present our testbed system developed to demonstrate our WLSN architecture. Section 6 highlights performance advantages of proposed WLSN architecture over the difficulty of the current WMSN. Finally, Section 7 gives our conclusions and future work.

2. STATE OF ART

Many research labs have started exploring the use of multimedia in WSNs. Most of their work is related to variable bandwidth allocation for video traffic, real time communication protocol to deliver quality of service, image data compression and middleware support for distributed imaging and database applications.

Reference [2], presents a multi-tier heterogeneous camera based surveillance network called SensEye. They make use of low fidelity camera sensors at the second tier to view an object and comparatively higher fidelity cameras at tier three to perform object tracking. IrisNet (Internet-scale Resource-Intensive Sensor Network Services) [3] is a heterogeneous wireless multimedia sensor network platform, which allows the user to perform queries over the video sensors distributed world wide. Another multi-tier wireless multimedia sensor testbed [4] is deployed at Broadband and Wireless Networking (BWN) Laboratory in Georgia Tech which makes use of scalar sensors, CMOS based camera nodes, medium-end video sensors are based on Logitech webcams interfaced with Stargate platforms and pan tilt cameras installed on a robotic platform.

Most of these multimedia wireless sensor network test-beds make use of CMOS based cameras such as Cyclops [5] and CMUcam3 [6] at a lower tier and Logitech webcams for high end image processing at the next higher tier. Stanford MeshEye [7] mote is smart camera architecture developed for distributed intelligent surveillance. It is used to determine the position, range and size of moving object. The platform is equipped with low resolution VGA camera module CC2420 transceiver for wireless communication. A new wireless camera vision system [8], is developed using two independent VGA color sensors on the same mote. The two cameras viewing the same scene from different view points can construct a high performance surveillance system using a onboard video analysis processor, 8051 microcontroller and IEEE802.15.4 enabled transceiver. The paper [9], describes a tiny CMOS-based single chip sensor of size less than 5mm on a side. It consists of a 320 * 240 pixel array and a radio module for communication. This platform is designed for biomedical applications.

A real time network simulator (RTNS)[10] based on NS2 provides a multi vision simulation environment based on the models of the architecture components such as real time OS, IEEE 802.15.4 based communication stack and imaging tasks used in the framework presented in this paper. Wireless Image Sensor Network Application Platform [11] is provides MatlabTM based simulation library for image sensors and wireless motes to investigate applications and algorithms for wireless image sensor network.

However, all the image based wireless sensor network platforms mentioned above are based on traditional two-dimensional image processing for object detection and tracking. In [12], Pagano et al have mentioned the communication bottleneck problem in a two-dimensional image processing node. Zheng and Sinha [13] have explored the use of line camera sensors in wired communication using high end video cameras. In contrast, in this paper, we will systematically investigate WLSN architecture with low end cameras in WSN under stringent resource limitations.

3. LINE SENSOR BASED ARCHITECTURE

3.1 Software-Based Line Sensor

A hardware-based line sensor capable of capturing one-dimensional image for high end applications are typically very expensive. It is not suited for our embedded systems to be deployed in an ad hoc manner under harsh environment conditions.

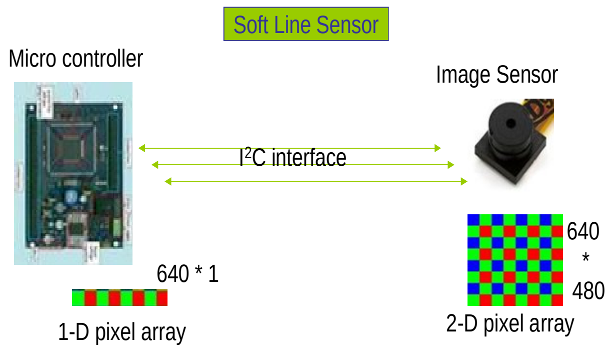


Figure 1. Flex - Camera Interface.

Instead, we propose an idea of software-based line sensor and achieved this by using some popular and inexpensive two-dimension image sensor such as HV7131GP. Our approach not only effectively realizes a software enabled line sensor for WLSN, but also provides us with the flexibility to obtain either a two or one dimensional image *on demand* and *online* depending on the application requirement in the same framework.

Figure 1 displays the hardware setup for software-based line sensor. The CMOS camera module of HV7131GP is able to capture the image at the maximum resolution of 640*480 pixels. This image is however a two dimensional image. This image is stored within the internal memory of the camera. This camera module has a feature which allows the microcontroller to select the desired width and height of the image. The camera module only transfers the selected part of the image from its internal buffer.

In order to obtain line image, the microcontroller selects the desired height of the image to be equal to one. This feature can be varied during the run time, thus enabling the application to run either in two- or one- dimensional mode of operation depending on the application requirement.

3.2 Principles in Generating Line Images

The proposed software-based line sensor extracts a line from the two dimensional image retrieved by the camera module. This is equivalent to focusing a sampling line (L) in the Field of View (P). A continuous projection of this sampling line will result in capturing a single line over the focused plane. Any static object in (L) will appear as part of the background, hence it cannot be detected. The continuous projection of the sampling line will capture consecutive portions of an object at different instants when it moves across the plane of sight as shown in Figure 2.

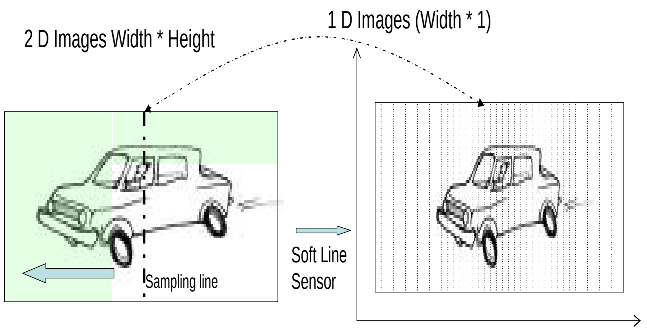


Figure 2. Line Sensor Image to Construct a Temporal View.

An effective scheme to capture all the details of the moving object depends on:

1. The projection of sampling line and the moving direction of the object.
   The camera focus and the sampling line will form a plane of sight within the focal distance of the camera. In order to capture the shape of an object in the moving direction V, the moving direction, n, of the line must satisfy the following condition:

   $V \cdot n \neq 0$; where n is a normal to the plane of sight in the 3D space.

   This means that the object should not move in a direction parallel to the plane of sight or the direction of line. This restriction shows that there is one specific direction of a moving object which cannot be captured with a given camera alignment. We can resolve this problem using a WSN based multi vision system to align the camera along the three basis orthogonal vectors in a 3D space.
2. A high sampling rate at which the sampling line is projected .

The rate at which the sampling line is projected is very important to capturing the moving object. If the sampling rate is too low it might miss an object altogether. On the other hand a fast sampling rate would replicate the same object portion. A proper sampling rate depends on the length and the velocity of the moving object assuming that the previous line sensor alignment condition is satisfied.

$$L_o = \frac{f \cdot L_i}{60 \cdot v}$$

where: L_o is the line sensor captured object length;

L_i is object length project on image frame

f is the line sensor sampling rate

v is the object speed.

As it can be seen from the above equation there exists a sampling rate for a particular object speed which will maintain the object resolution.

This sampling rate places an additional timing constraint on the application. The image processing and communication task needs to be finished before the next sampling period. Otherwise either the sampling line has to be skipped or the processing tasks have to be terminated. We address this problem by proper selection of periodic tasks with priorities using the services provided by ERIKA.

4. LINE SENSOR BASED ONE DIMENSIONAL IMAGE PROCESSING ALGORITHMS

We devise a suit of low-complexity algorithms for processing line images generated by our software enabled line sensor for object detection in WLSN. Everything static appears as a background and hence eliminated from the images. These algorithms are only able to detect moving objects whose speed is within the limits set by the sampling rate of line images. To address the resource constraints in sensor nodes, we make the assumption that a sensor node (i.e., end device) is only able to store a single array of one dimensional line where as the base station can store multiple lines.

Consider the image buffer at the base station, which is a dynamic circular FIFO two-dimensional buffer, with the size of 640 (one dimensional image width) * 480(window buffer height), for example. Every new line is added at top thus overwriting the oldest line in the window of the buffer. This window is continuously displayed giving an impression of a scanned image.

4.1 Background Image

The first algorithm is to calculate the camera sight background on every input line image (referred to as Algorithm 1). This background one dimensional image is maintained in a separate buffer of 640 bytes. In order to identify a line as a background every pixel is compared with the buffered line. Every pixel in the input line is matched with the corresponding pixel in the buffered line. If each individual pixel difference is within the specified threshold then the input line is considered to be a background image. The new background buffer value is the average of the two lines. The average value helps to maintain a continuously updating background with an O(n) complexity.

```
Algorithm 1: Background image formation
Collect the input line input (i)
If ∀i,: 0<i< WIDTH it follows:
        || bGavG (i) – input(i) || < threshold_1
then:
        bGavG(i) := (bGavG (i) + input(i))/2;
```

4.2 Foreground Image

The foreground image is computed by subtracting every input image with the average background image.

The line is first compared against the background image to identify it as a foreground or background line (as described in Section 5.1). Once, it has been identified as belonging to the foreground image, the difference image between input and background is compared to a threshold. The pixel having intensity greater than a threshold is projected as a foreground object pixel with intensity set to 255 (the background is assigned a null intensity). This threshold allows generating a binary image with noise removal.

```
Algorithm 2: Foreground image formation
Collect the input line input (i) recognized as "NOT" background
in Algorithm 1
If ∀i,:  0<i< WIDTH it follows:
        $|input(i) – bGavG(i)| > threshold_2$
then:
        input(i) = 255;
else:
        input(i) = 0.
```

The binary foreground image simplifies further processing of moving objects by eliminating unwanted data. It is also easier to compress the image data since most of the pixels have null intensity. Depending on the mode of operation the data transmission can be further reduced by transmitting only the foreground pixels.

4.3 Moving Object Boundary Extraction

The object boundary detection is a crucial step to detect and recognize the type of moving objects. The boundary extraction has not been implemented on the end device. Due to its memory limitation end device is able to store only one image line at a time. This makes the separation of the object boundary from the object body difficult since it uses global information spanning multiple lines.

Since the base station is able to maintain a moving window of image lines with a height of 480 lines, it is possible to process multiple lines. However, this is different from a two dimensional image processing where the image data does not change for the duration of image processing, whereas in our implementation the processing needs to be finished before a new line enters the buffer. This imposes a real time deadline for the algorithm. Because of this restriction, we limit our processing to portion of moving window so as to finish the computation before its deadline.

After the foreground is separated from the background in the previous step, all the object pixels are assigned an intensity value of 255. The boundary extraction algorithm uses a connected labeling to identify 8 neighbor pixels. A pixel having 8 connected neighbors whose value is greater than 0 is identified as an interior object pixel, whose intensity is changed to a lower value. This process requires only three image lines at a time from the moving buffer.

Thus the output of this process is a moving image window having background pixels with intensity 0, boundary pixels with intensity 255 and interior pixels with intensity greater than 0 and less than 255.

4.4 Moving object detection and reporting

The moving object detection works both at the end device as well as the base station. The algorithm at the base station is a simple extension of boundary extraction. Once the boundary has been extracted a new object is recognized and this datum is maintained along with other important details to maintain a state of all detected objects.

The algorithm at the end device is supposed to work on single image lines, which requires it to maintain vector information about the previously processed lines.

A line image after a certain number of background lines, which is detected to have a foreground object, is identified as a start of object. This line could have several connected segments. A set of connect pixels with count greater than a segment threshold is identified as a connected segment. The information vector maintains knowledge on these segments. A set of lines having consecutive connected segments aligned with the previous line and having a set count greater than an object threshold is identified as an object.

The segments and objects not fulfilling the threshold criteria are discarded as noise. Thus, this simple algorithm with a single line and vector information on previous lines is able to detect a moving object.

Depending on the mode of operation the end device send either a simple report on detected object or it can send the object lines thus informing the base station not only about a detected object but also its image.

5. TESTBED SYSTEM

We have developed a testbed including all hardware and software components for our proposed WLSN architecture. This system consists of a flexible hardware architecture designed to adapt to users' needs for constructing an embedded system. The software components are suited for a distributed application composing; a real time operating systems, device drivers and wireless stack supporting IEEE 802.15.4 specifications.

5.1 Hardware Platforms

Figure 3 shows the overall hardware design for our line sensor node.

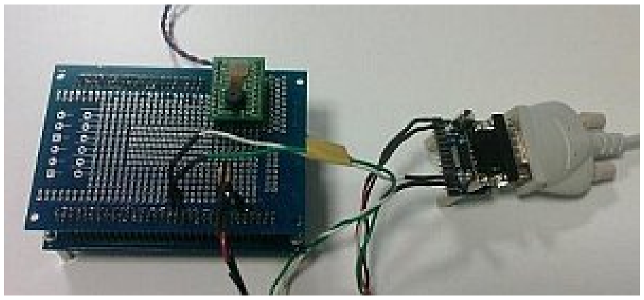


Figure 3. Line Sensor End Device Setup.

The flexible design of this module allows the developer to add components based on his choice. This setup consists of Flex boards with a HV7131GP camera attached to a breakout board and a serial to TTL converter. The board is connected to the base station (personal computer) using a serial to USB converter. The flex board uses UART to communicate with the base station. Here we describe the selected hardware components used in our implementation of software based line sensor imaging.

5.1.1 Flex board

Flex [14] is an embedded board which can be used by all the developers who want to fully exploit the potential of the latest

Figure 4.

CC2420 Transciever.

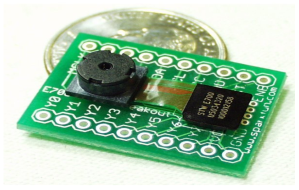


Figure 5.

HV7131GP CMOS Camera.

Microchip micro-controllers: the dsPic family. Flex is born as a development board where to easily develop and test real-time applications.

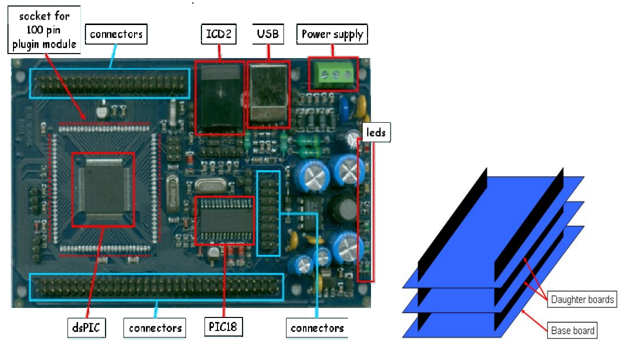


Figure 6. Flex Board.

The basic configuration of a flex device is made by the Base Board only. The flex Base Board mounts a Microchip dsPic micro-controller, and exports almost all the pins of the micro-controller. The user can easily connect the desired components to the dsPic ports in order to build the specific application. As depicted in Figure 6, several daughter boards can be connected in piggyback to the Flex Base Board. The daughter boards have different features and they can be easily combined to obtain complex devices.

This architecture includes dsPic33F from 16 bit family of dsPic microcontrollers. The CPU operates at a 40 MIPS (i.e. 40 million instructions per second) internal clock frequency and 30 KB of internal RAM memory. This CPU speed and memory limitation imposes severe constraints on the line sensor based imaging application which demands heavy computation and storage requirements.

5.1.2 *HV7131GP Camera Module*

HV7131GP [15] (Figure 5) is an integrated single chip CMOS image sensor with certain image processing capabilities such as gamma correction, color interpolation, auto exposure control and auto white balance. It provides different levels of resolution with maximum achievable being 640*480 pixels. This feature allows setting the resolution depending on the desired sampling rate and available memory on the end device.

The chip allows for a maximum adjustable frame rate of 30 f/s. In our application we have used 8 bit gray scale image, although the chip provides 8bit and 16 bit RGB and YC_bC_r output format. The camera module allows capturing the image by setting its width and height along with its frame coordinates. This feature enabled us to implement one dimensional line sensor based feature using a traditional two dimensional image senor. The sensor uses I^2C based communication channel to send and receive commands and transfer bit map image to the end device.

5.1.3 *Chipcon CC2420*

The CC2420 [16] (Figure 4) is a single chip IEEE 802.15.4 compliant RF transceiver operating in the 2.4 GHz band with an effective data rate of 250 kbps.

The transceiver hardware supports many of the features required to implement an IEEE 802.15.4 protocol stack. These features include: clear channel assessment, link quality indication and support for buffered packet handling. The configuration, command and data communication is accesses via a SPI interface between the microcontroller and the transceiver.

5.2 Software Components

This section describes the layered architecture consisting of the drivers, operating systems, wireless communication stack, imaging component and the line sensor based application for object detection.

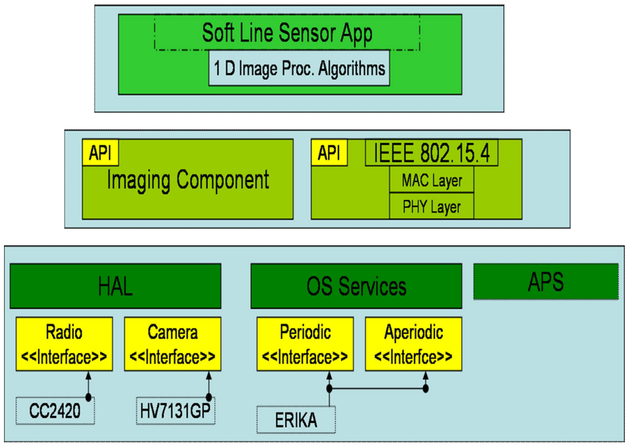


Figure 7. Layered Architecture.

The above figure shows the interaction among different subsystems of this software architecture. The Hardware Adaptation Layer (HAL) provides a hardware independent interface for using the transceiver and camera module. In our design we make use of CC2420 and HV7131GP device drivers to implement the features requested by radio and camera HAL interface.

The services required from the kernel are related to external event handling and multi-thread programming. We namely want to run periodic and aperiodic activities by means of timers, alarms, events and tasks instantiated in the service and user layers.

In the following subsection the adoption of ERIKA, a multi threaded real time Operating System suited for time critical image processing based distributed application, will be described in details.

5.2.1 *Erika*

Erika provides an abstraction of the machine hardware and is in charge of reacting to events and handling access to memory, CPU, and hardware peripherals. Especially in constrained hardware devices like those of sensor boards, the effectiveness in the OS paradigms largely affects the response in the target application. The execution model is the key factor differentiating the many solutions in existing OS for WSN. ERIKA Enterprise [17] RTOS is a multi-processor real-time operating system kernel, implementing a collection of Application Programming Interfaces (APIs) similar to those of OSEK/VDX standard for automotive embedded controllers. ERIKA is available for several hardware platforms and it introduces innovative concepts, mechanisms and programming features to support micro-controllers and multicore systems-on-a-chip. ERIKA features a real-time scheduler and resource managers, allowing the full exploitation of the power of new generation micro-controllers and multicore platforms. Tasks in ERIKA are scheduled according to fixed and dynamic priorities, and share resources using the Immediate Priority Ceiling protocol. Interrupts always pre-empt the running task to execute urgent operations required by peripherals. The wireless communication protocol makes use of the radio interface HAL and OSAL. Our current design makes use of ERIKA and cc2420 as the underlying implementation for the radio and OS modules.

5.2.2 OpenZB

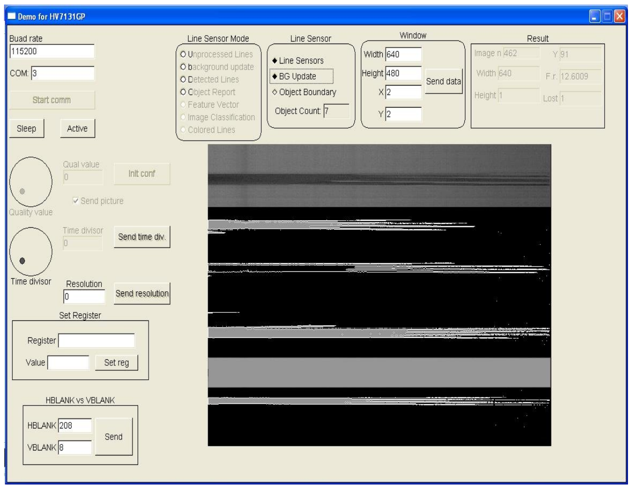
OpenZB[18] is an implementation of IEEE 802.15.4 protocol stack. This implementation can operate both in unslotted and slotted CSMA/CA mode described in the standard specifications. Since the software design is developed over ERIKA operating system, the protocol stack makes use of the software abstractions such as alarms for effectively providing the timing behavior in slotted mode of operation; the kernel scheduling policy (driven by static priority settings) handles the time critical services of the wireless stack. The libraries support the generation of the MAC superframe and provide the slotted CSMA/CA access mechanisms. The protocol services have been mapped to tasks having reserved a set of priorities for network-related use only. OpenZB is able to provide timing guarantees over the network by allocating slots to different nodes in Guaranteed Time Slots. This behavior is crucial for distributed vision applications which need to detect and track any desired object.

The application support (APS) provides utilities to handle communication, PC display, string handling and memory management. The imaging component provides the necessary functionalities to configure and retrieve one- or two-dimensional images using I^2C interface between the microcontroller and image sensor. The component can handle the configuration at the run time. The configuration features include: adjusting the resolution, setting image height and width, changing the frame rate, sampling rate and additional commands to enable different operation modes of line sensor in the WLSN applications for line imaging processing to detect moving objects.

5.2.3 Base Station Application for user interaction

This is a GUI based application for the end user to configure the camera setting and keep track of the objects. The user can adjust the image mode to either two dimension image or line sensor based images. When used under line sensor mode the user can select to handle the image processing features either at the end device or at the base station. This application also enables to visualize lines with or without background elimination, to view only the images where an object is being detected or only a report on the detected object. These different modes of operations along with the image processing algorithms are explained in the next section.

The base station uses a RS 232 – TTL interface to communicate with the device using a serial to USB converter. The device sends the images to the base station depending on the configuration set by the base station. These configuration parameters could be changed by the end user at the run time via the GUI.


Demo for HV7131GP

Buad rate: 115200
COM: 3

Line Sensor Mode: Unprocessed Lines / background update / Detected Lines / Object Report / Feature Vector / Image Classification / Colored Lines

Line Sensor: [✓ Line Sensors] [✓ BG Update] Object Boundary
Object Count: 7

Window: Width 640, Height 480, X 2, Y 2

Result: Image n 462, Y 91, Width 640, F.r. 12.6009, Height 1, Lost 1

Qual value: 0
[✓ Send picture]
Time divisor: 0
Resolution: 0

Set Register: Register , Value

HBLANK vs VBLANK: HBLANK 208, VBLANK 8

Figure 8. Base Station User Interface.

6. PERFORMANCE OF WLSN

6.1 Traditional Architecture Bottlenecks

The computation, communication and storage limitations of embedded devices present various limitations in current wireless multimedia sensor networks. The WLSN is designed to overcome those bottlenecks.

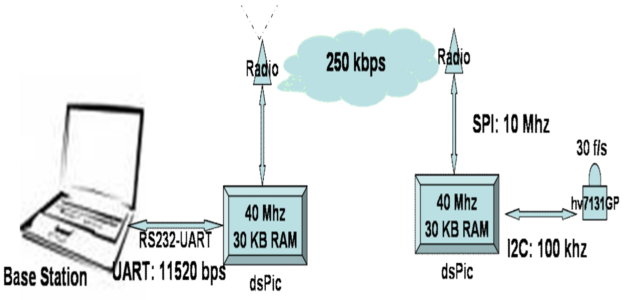


Figure 9. Communication Interfaces.

Figure 9 depicts various bottlenecks in the senor network architecture along the image data flow from the point of its sensing to its final destination (i.e., base station). The camera module (having a maximum frame rate of 30 f/s) transfers the image (either one or two dimensional) over an I^2C interface with its maximum bandwidth of 1 MHz. This bottleneck restricts the sampling rate of the images. The sampling rate of the image is important to capture the details of a moving object. This relationship between the speed of the object and sampling rate will be described in the next section. The dsPIC microcontroller operating at 40MHz with a 30KB RAM size forces to use variations of image processing algorithms operating on one dimensional image buffer, i.e. the algorithms need to remember the information present in the previous lines while maintaining only one line in the buffer at a time.

The microcontroller communicates to a remote master node (playing the role of network coordinator) the processes or unprocessed image (depending on the selected mode of operation) over a wireless medium using the IEEE802.15.4 based transceiver having a maximum bandwidth of 250 kbps. Since this medium is shared among many nodes it restricts the amount of information which can be transferred over the medium.

In order or transfer less information, the end device needs to implement complex image processing algorithms to extract useful information. This computation/bandwidth tradeoff is handled by the end user by selecting the appropriate mode of operation. Finally, the received information is transferred to the base station via a RS232 interface with a maximum bandwidth of 115200 kbps. Usually, the base station is a comparatively high end machine (e.g. a PC) capable of storing multiple images and executing complex image processing algorithms.

6.2 Operation Modes of Line Sensor

To optimize the performance of WLSN with given resource limits on sensor nodes and network bandwidth for a particular application at hand, we exploit different trade-offs between bandwidth and computation in the WLSN by the means of different operation modes of line sensor. The table below lists five different modes we investigated and their comparative performances with respect to storage and computation on different line sensor operation modes.

Table 1. Operating Modes

Operation Mode	Storage (No. of bytes)	Computation complexity
A: (2-D) Images	Width * Height	NA
B: 1D Images without processing	Width * 1 (grayscale image)	NA
C: 1-D Images with background update	Width * 1 (binary image)	O (Width)
D: 1-D Images for detected parts	Width * 1	O (Width)
E: Object Report	Boolean (1 byte)	O (Width)

The simplest mode (A) is to receive a 2-D image from the camera module. Due to the RAM limitations, the maximum size of 2-D image that can be stored is 160 * 120 bytes. In this mode the image is retrieved and displayed at the base station with a constant frame rate. This mode of operation requires a large storage and bandwidth capacity. Since the image processing is performed at the base station computation limitation is ignored. The other operation modes work on one dimensional image captured from the camera module. The basic mode of operation on line images (B) is just to receive the image by setting appropriate width and height equal to one. The end device transmits this image without any processing, leaving it for the base station. Because of the size reduction, the (microcontroller) can receive the 1-D images at a higher rate and (within some limitations) can store them in main memory: with 640 pixel resolution, a grayscale line image with 8 bits per pixel (0 to 255 intensity levels) occupies less than 1Kbyte in RAM so that it is feasible to store a sequence of 20 lines..

The next mode (C) involves some processing on the line images at the end device before transmitting it over the wireless medium. This image processing although involves minimal amount of computation on end devices reduces the bandwidth requirement of wireless channel. This is significant in WSN where multiple nodes need to exchange information on the environment over the limited bandwidth.

The simplest of this computation involves separating foreground from background and representing the foreground as intensity 255 while background as 0, thus creating a binary image displaying only moving objects. The next mode of operation (D) sends the line sensor data only when an object is detected, i.e. the data segments displaying the object and finally in mode (E) the end device runs the object detection algorithm locally and sends a “binary” report only when an object is detected. The following section describes the algorithms involved both at end device and base station while operating in different line sensor modes As we move to a higher operation mode, the bandwidth requirement is significantly reduced at the expense of more computation on each end device. The operation mode selection can be delegated to the final user or dynamically automated depending on the object speed, available bandwidth, processing speed, etc to operate at the optimal level.

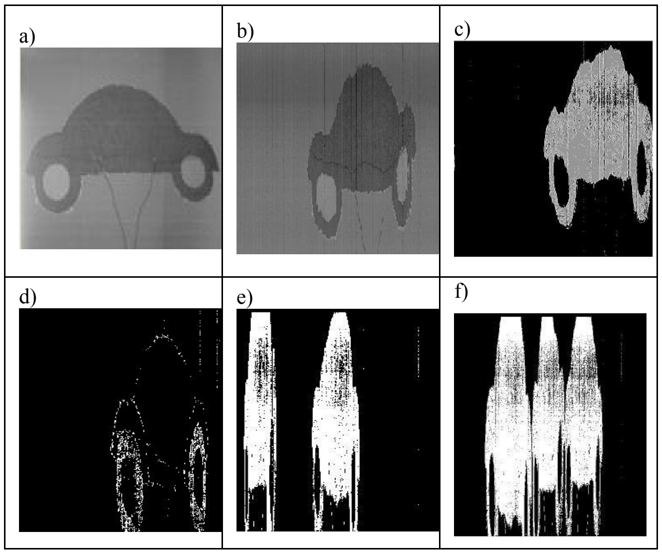


Figure 10. Line Sensor Images under different operation modes.

Figure 10 shows the output images when operating under different line sensor modes. Figure 10 (a) is a 160 * 120 byte grayscale image. (b) is a one dimensional grayscale image of a moving car. Figure 10 (c) shows a background updates binary image. This is obtained by subtracting the average background image from the current line to display only a binary foreground moving object. Figure 10 (d) shows only the moving object boundary. Figure 10 (e) shows the result of background updating performed at the end device. It shows the result of two cars passing after one another with background lines in between them carrying no information. Figure 10 (f) displays the result of image processing at end device. In this case only detected lines are transmitted to the end device. The background lines which do not carry any information are not transmitted, thus saving the wireless bandwidth. Operation mode (E) of object detection does not display any image. It only sends a report of the center of mass of the detected object, thus maintaining the count, time and position of the detected object at the base station. The end devices do not maintain any state or past history about the images.

6.3 Advantages of WLSN Architecture

The proposed WLSN architecture with the line imaging has several advantages over the traditional two-dimensional imaging based WSN.The smaller image data size significantly saves the node memory as well as the network bandwidth. Since the amount of data to transmitted and received is reduced this also greatly helps conserve energy spent on communication. Most of the simple algorithms on line sensors devised try to achieve the goal of object detection with a linear complexity. The minimal image data and commutation thus reduce the end-to-end delay between the end devices and the base station as well.

The line images are able to work efficiently under the hardware bottlenecks described earlier. Because of this the end device is able to sample the image at a much higher rate than the two dimensional image. This enables WLSN based applications to capture important events which could have been missed otherwise because of the lower sampling rate with two-dimensional images in the current WSN for visual surveillance.

7. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a novel wireless line sensor network architecture with software-based line sensors for visual surveillance. We have also devised a suit of algorithms for line image processing. To verify our idea, a testbed is developed on a real time operating system working on a customizable board design installed with a camera module and transceiver. The OpenZB protocol stack is used to achieve a distributed multivision system in the WLSN testbed. Furthermore, the proposed WLSN enables to operate in different modes of line sensors to achieve an optimal tradeoff between communication and computation for any given application situation.

In the future, we will study our WLSN in distributed applications over several nodes in a multihop network with CSMA v/s GTS based medium access communication using our network stack. We also plan to develop an analytical model to represent the communication, computation usage considering the node power and environment interference to operate the system in the most appropriate line sensor mode.

ACKNOWLEDGMENTS

We thank Claudio Salvadori and Christian Nastasi on providing the initial version on camera driver and application.

REFERENCES

[1] IEEE 802.15.4 Standard-2003, “Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)”, IEEE-SA Standards Board, 2003.

[2] P. Kulkarni, D. Ganesan, P. Shenoy, Q. Lu, SensEye: a multi-tier camera sensor network, in: Proc. of ACM Multimedia, Singapore, November 2005.

[3] S. Nath, Y. Ke, P.B. Gibbons, B. Karp, S. Seshan, A distributed filtering architecture for multimedia sensors, Intel Research Technical Report IRP-TR-04-16, August

[4] Wireless Multimedia Sensor Testbed. http://www.ece.gatech.edu/research/labs/bwn/WMSN/testbed.html

[5] M. Rahimi, R. Baer, O. Iroezi, J. Garcia, J. Warrior, D. Estrin, M. Srivastava, Cyclops: in situ image sensing and interpretation in wireless sensor networks, in: Proc. of the ACM Conf. on Embedded Networked Sensor Systems (SenSys), San Diego, CA, November 2005.

[6] Carnegie Mellon Univ., CMUcam3 datasheet version 1.02, Pittsburgh, PA, Sep. 2007.

[7] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan, MeshEye: A hybrid-resolution smart camera mote for applications in distributed intelligent surveillance, in Proc. Int. Conf. Inf. Process. Sensor Netw. (IPSN), Cambridge, MA, 2007, pp. 360–369.

[8] R. Kleihorst, B. Schueler, A. Danilin, and M. Heijligers, Smart camera mote with high performance vision system, in Proc. ACM SenSys Workshop Distrib. Smart Cameras (DSC), Boulder, CO, Oct. 2006.

[9] S. Itoh, S. Kawahito, and S. Terakawa, A 2.6 mW 2 fps QVGA CMOS one-chip wireless camera with digital image transmission function for capsule endoscopes,[in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2006.

[10] Real Time Network Simulator. http://rtns.sssup.it/RTNSWebSite/RTNS.html

[11] WiSNAP: A Wireless Image Sensor Network Application Platform. S. Hengstler and H. Aghajan, 2nd Int. Conf. on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom), March 2006.

[12] P.Pagano, C.Nastasi, Y.Liang, The Multivision problem for Wireless Sensor Networks: a discussion about Node and Network architecture. In International Workshop on Cyber-Physical Systems Challenges and Applications (CPS-CA08). Proc. of the DCOSS 2008 conference., Santorini island, Greece, June 2008. Invited talk.

[13] J. Y. Zheng, S. Sinha, Line cameras for monitoring and surveillance sensor networks, ACM Conf. Multimedia 07, 433-442, Augsburg, Germany, 2007

[14] The Flex board. http://www.evidence.eu.com/.

[15] CMOS Image Sensor with Image Processing. HV7131GP www.globaltec.com.hk/databook/hynix/Hyca3_V20.pdf

[16] Low Power RF Transceiver CC2420. http://focus.ti.com/docs/prod/folders/print/cc2420.html

[17] E.R.I.K.A. http://erika.sssup.it/.

[18] P.Pagano et al, ERIKA and open-ZB: an implementation for real-time wireless networking. SAC 2009, 1687-1688.