ON THE DESIGN AND IMPLEMENTATION OF WIRELESS MULTIMEDIA SENSOR NETWORKS

Thomas Davis, Qing Pang, Gordon W. Skelton, Natarajan Meghanathan
Jackson State University, Jackson, MS 39217, USA
natarajan.meghanathan@jsums.edu

ABSTRACT

Wireless Sensor Networks (WSN) mainly deal with scalar data such as temperature, humidity, and light which are very suitable for low rate and low power IEEE 802.15 based networking technology. The commercial off-the-shelf (COTS) CMOS camera has fostered researchers to push WSN a step further. The unique properties of multimedia data delivery pose fresh challenges for resource constrained sensor networks. Transmitting raw data is very costly while limited processing power prevents sophisticated multimedia processing at the sensor nodes. Wireless sensor networks offer an attractive choice for low cost solutions for transmitting data wirelessly to a database to be evaluated. Wireless networks of visual sensors have recently emerged as a new type of sensor-based intelligence system. The goal of the visual sensor network is to provide a user with visual information from an arbitrary viewpoint within the monitored field. Wireless networks in combination with image sensors open up a multitude of previously unthinkable sensing applications. In an on-going project, we are designing and implementing a sensor node with a camera which would be capable of acquiring still images, transfer the data onto a personal computer through wireless communication, and store the image on a personal computer. This paper explains the process of capturing the raw image data with a camera sensor and the interfacing of the camera with the Overo Air computer-on-module (COM). Camera visibility and resolution will also be explained in this paper along with the procedure taken to configure the sensor node.

KEYWORDS

Wireless Sensor Networks, Multimedia, Camera, Overo Air Computer-on-Module, System Architecture

1. INTRODUCTION

Wireless sensor networks have drawn the attention of the research community in the past few years. This growing interest can be largely credited to new applications enabled by large-scale networks. Small devices are capable of harvesting information from the physical environment, performing simple processing on the extracted data, and transmitting it to remote locations. Most deployed wireless sensor networks measure scalar data such as temperature, pressure, humidity, or the location of objects. In general, most of the applications have low bandwidth demands and are usually delay tolerant.

While the traditional wireless sensor networks consist of low-bandwidth sensors with limited capabilities, camera sensor networks can provide visual verification, in-depth situational awareness, recognition, and other capabilities. In recent years, wireless sensor networks have inspired tremendous research interest in diverse application fields such as structural health monitoring, environmental monitoring, and military and security surveillance. A typical wireless sensor network consists of sensor nodes, each equipped with various kinds of sensors, deployed over a geographical region of interest. Most of the applications are centered towards harvesting information from the physical environment, performing a simple processing on the extracted data and transmitting it to remote locations.

A new direction in wireless sensor network application design centers on the idea of enabling the network to learn the behavior of the trend in the environment rather than merely making measurements and reporting about a single event or information of interest. A Wireless Multimedia Sensor Network (WMSN) is defined as a network of wireless embedded devices that allow retrieving video or audio streams, still images, and scalar sensor data from the

DOI : 10.5121/ijngn.2010.2303

physical environment. A WMSN ties the notion of wireless sensor networks and distributed smart cameras [1].

Wireless multimedia sensor networks will not only enhance existing sensor network applications such as tracking, home automation, and environmental monitoring, but they will also enable several new applications such as multimedia surveillance sensor networks. Wireless video sensor networks will be composed of interconnected, battery-powered miniature video cameras, each packaged with a low power wireless transceiver that is capable of processing, sending, and receiving data. Video and audio sensors will be used to enhance and complement existing surveillance systems. Large-scale networks of video sensors can extend the ability of law enforcement agencies to monitor areas, public events, private properties and borders. Multimedia sensors could infer and record important information about an activity such as car theft, car accident, or traffic violation, and make live video/audio streams or reports available for future studies. A WMSN could be deployed to monitor car traffic in big cities and on highways and deploy necessary services that offer traffic routing advice to avoid traffic congestion. In addition, smart parking advice systems will allow monitoring for available parking spaces and provide the drivers with automated parking advice, improving vehicle mobility in urban areas. Also, multimedia sensors may monitor the flow of vehicular traffic on highways and retrieve information such as average speed and number of cars. Sensors could also detect violations and transmit video streams to law enforcement agencies to identify the violator.

The remainder of this paper is organized as follows. In Section 2, we briefly explain the components, their functions, and their capabilities with regards to a WMSN. Section 3 describes the overall system architecture of our proposed WMSN development platform and also discusses the power distribution and management of the sensor network. Section 4 compares the camera's resolution vis-à-vis the display resolution. Section 5 discusses the software used and Section 6 explains the sequence of steps to be performed to implement the proposed WMSN architecture. Finally, Section 7 summarizes the main aspects of the application development platform introduced and outlines directions for further work.

2. MAIN COMPONENTS

Some of the components to build a WMSN can be very expensive and difficult to work with or program due to the complicated algorithms and required hardware construction. We had to decide on the type of microprocessor to use to run our node along with a camera with good resolution that could capture effective images. We considered two microprocessor modules: the ARM7 microprocessor and the Overo Air computer-on-module. The Overo (shown in Figure 1) seemed like a better choice due to its extensive wireless capabilities that the ARM7 did not have. With regards to the camera, we researched for a camera that had an existing driver rather than creating our own driver for the camera module. Upon researching, we came across a camera module (shown in Figure 2) that supports the Overo Air computer-on-module. It connects to the Overo Air through a J5 flex circuit cable connector, making it simpler to communicate between the two devices.

The components that we will be using are very cost efficient and easy-to-use devices. We will be working with an inexpensive Overo Air computer-on-module along with a Summit expansion board (refer to Figure 3), an expansion camera board, and a USB Wi-Fi dongle. The

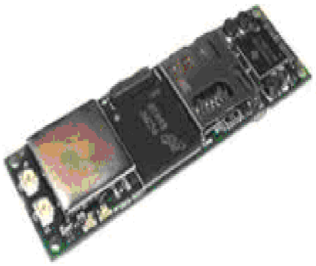


Figure 1: Over Air COM Module

Figure 2: Camera Module

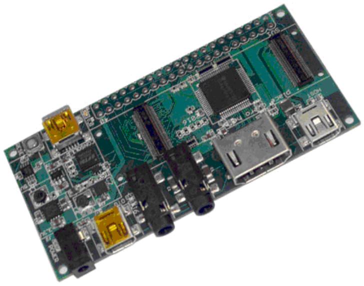


Figure 3: Summit Expansion Board

Overo Air is a tiny computer module made by Gumstix that can communicate via wireless or Bluetooth. The size of the computer is 17mm x 58mm x 4.2mm making it one of the smallest, efficient, easy-to-use processors around. It is considered to have high performance and low cost for a product of its size. The module uses the OMAP 3503 Application Processor with ARM Cortex-A8 CPU. Further information about the Overo Air COM can be found at [2]. The Summit is an expansion board also made by Gumstix to help support the features of the Overo Air COM. It has a power supply to help support itself and the Overo. It also has USB jacks for serial programming and for operating in Host or Device mode. The size of the board is 80 mm x 39mm. More information about the Summit expansion board is available at [3]. The camera module made by e-con Systems is a 3.2 Megapixel camera that works as a plug-and-play for the Overo Air COM. It has max resolution at QXGA (2048x1536) and is able to catch up to 30 frames per second. The size of the module is 17mm x 58mm. More information can be found at [4]. The USB Wi-Fi dongle will be used for wireless communication to the host computer. It has a high speed transfer date rate up to 54 megabits per second. It has data encryption with WEP and WPA and also can transmit and receive date from up to 300 meters indoors and 800 meters outdoors. The USB wireless adapter also supports and is compatible with Microsoft, Apple, and Linux operating systems. More information pertaining to the USB Wi-Fi dongle can be found at [5].

3. System Architecture

Figure 4 presents a summarized view of the circuit block diagram for our on-going project. As one can see, communications are made between the Overo Air COM with the camera module and the Summit expansion board. The Overo Air COM and the Summit expansion board are connected via a 70-pin connector. The camera module and the Overo Air are connected using a flex circuit cable connector. Power will be supplied to all components for proper functionality through the Summit expansion board. After repetitive attempts of trying to get the onboard Wi-Fi to work, we decided that it would be best we just attach a USB Wi-Fi dongle to the Summit expansion board for wireless communication. The Overo Air is the master component of the whole system. It communicates with all the components attached to it and also to the computer through a USB cable. The USB cable allows communication with the Overo's serial port console. The camera module captures the images and sends the raw data to the Overo Air COM to be processed and stored on the micro SD card. The card can then be read by the multi-media application, which can then send the information (wireless communication) to the computer for further processing and analysis.

The power supply management system proposed for our system architecture consists of some NiMH batteries and a battery holder, which secures the batteries in place to hold a charge. A 5V power supply will be plugged into the Summit expansion board and will be used to provide power and current along to the Overo Air COM and the camera module. Some of the power is also sent to the TPS65960 power management Integrated Circuit on the Overo Air COM where it can be evenly distributed along the board.

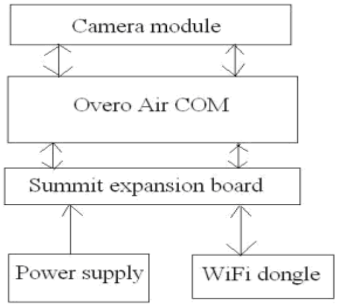


Figure 4: Circuit Block Diagram of the System Architecture

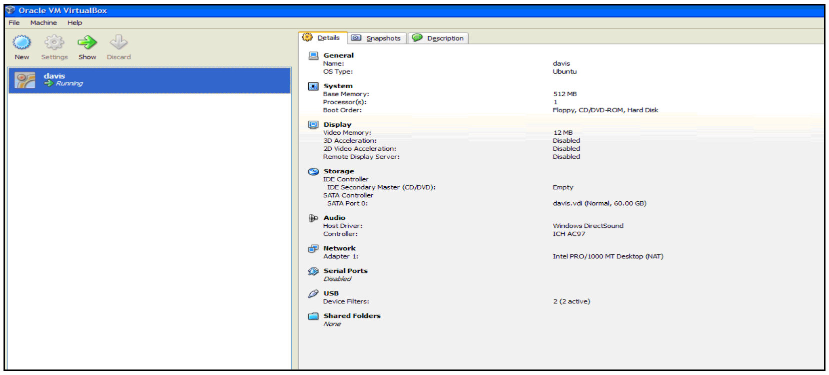
4. Camera Visibility vs. Display Resolution

Generally, as one views subjects from greater and greater distances, the magnitude of the details captured on the subject begins to fade away. The distance between the details is measured by the angle between them as one views them. As the distance between the subject and the camera increases, the angle fades, thus the detail also diminishes. Lens quality is very significant due to the increasing number of megapixels found in today's digital cameras. Frequently, the resolution of digital photos is actually limited by the camera's lens and not by the resolution of the camera itself [6]. There are a few obvious factors to being able to detect resolution differences. These factors include the resolution of the screen being viewed, the size of the screen and the viewing distance. To be able to detect differences between resolutions, the screen must be large enough and one must sit close enough [7]. Resolution describes the camera's ability to distinguish between closely spaced elements with respect to the details. To explain this better, imagine that one has two displays, in front, of the same size. The first display has 1080 pixels resolution while the other display has 720 pixels. If one sits close enough to get the best detail from the 1080p display, one will also probably notice the pixels of the 720p display. On the other hand, if one sits far enough to get the best detail from the 720p display, one will no longer see the full details from the 1080p display; the image will look the same on both. Thus, in order to determine the correct size based on viewing distance, we must also take into consideration the resolution of the display [7].

5. Software

The software that we are working on for our project is based on the Open Embedded (OE) environment. It is a software framework (a screenshot is presented in Figure 5) that is used mainly on the Linux operating system and is mostly common for embedded devices. It supports many hardware architectures and also is easy to customize. It allows developers to create a complete Linux distribution for embedded systems. The software framework maintains and develops a collection of BitBake recipes to build the application package. It includes all the package dependencies, URLs to retrieve the source code from, configuration, compilation, build, install and remove instructions. The framework also stores the metadata for the package in standard variables.

The BitBake recipes consist of the source URL of the package, dependencies and compile or install options. During the build process, these recipes are used to track dependencies, cross-compile the package and pack it up, suitable to be installed on the device. It is also possible to create complete images, consisting of the root file system and the kernel. The Open Embedded environment requires several mandatory packages to be downloaded. For proper functionality, the OE metadata needs to be installed with different packages and remain up-to-date.



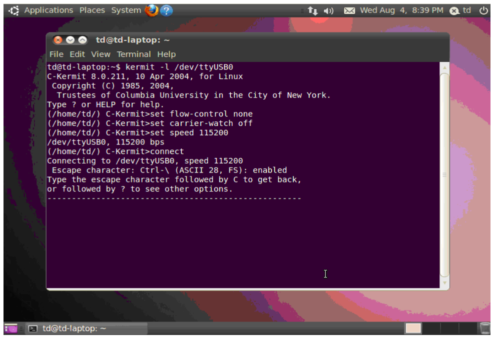
```
General
Name:                                   davis
OS Type:                                Ubuntu
System
Base Memory:                            512 MB
Processor(s):                           1
Boot Order:                             Floppy, CD/DVD-ROM, Hard Disk
Display
Video Memory:                           12 MB
3D Acceleration:                        Disabled
2D Video Acceleration:                  Disabled
Remote Display Server:                  Disabled
Storage
IDE Controller
  IDE Secondary Master (CD/DVD):        Empty
SATA Controller
  SATA Port 0:                          davis.vdi (Normal, 60.00 GB)
Audio
Host Driver:                            Windows DirectSound
Controller:                             ICH AC97
Network
Adapter 1:                              Intel PRO/1000 MT Desktop (NAT)
Serial Ports
Disabled
USB
Device Filters:                         2 (2 active)
Shared Folders
None
```

Figure 5: Virtual Box Application of the Open Embedded (OE) Framework

6. Procedure to Implement the Wireless Multi-media Sensor Network

In order for us to be able to access the Overo module and write programmable code for it, the operating system needed is Linux. If a user computer has been running Windows or any operating system other Linux, then we recommend the use of an application called Virtual Box. It is a multi-software virtual package that allows the user to be able to use any operating system including Windows, Linux, and Mac. Using the Virtual Box, we were able to set up a Linux Ubuntu operating system to work with the Overo Air COM.

The procedure to set up the Overo console session is somewhat easy; however, setting up the Open Embedded environment and coding for the environment is relatively complicated. First, to set up an Overo session (sample screenshot shown in Figure 6), a USB mini-B to standard-A cable should be connected between the computer and the Summit expansion board where the Overo Air COM is connected. Upon connecting to the Linux operating system, the terminal will be where most of the work is done. The terminal will be needed for most of the updates, downloads, and coding.



```
td@td-laptop:~$ kermit -l /dev/ttyUSB0
C-Kermit 8.0.211, 10 Apr 2004, for Linux
 Copyright (C) 1985, 2004,
  Trustees of Columbia University in the City of New York.
Type ? or HELP for help.
(/home/td/) C-Kermit>set flow-control none
(/home/td/) C-Kermit>set carrier-watch off
(/home/td/) C-Kermit>set speed 115200
/dev/ttyUSB0, 115200 bps
(/home/td/) C-Kermit>connect
Connecting to /dev/ttyUSB0, speed 115200
 Escape character: Ctrl-\ (ASCII 28, FS): enabled
Type the escape character followed by C to get back,
or followed by ? to see other options.
----------------------------------------------------
```

Figure 6: Setting up the Overo Console Session

Once the terminal has started, different packages that are needed for the installation of the Open Embedded (OE) environment will need to be downloaded and installed to make sure everything works properly. Downloaded packages include Python, GNU, BitBake, diffstat, perl, and Psyco

JIT Compiler. Linux has different commands to obtain the packages such as the *wget* and *git* commands. Also, four files are needed to boot the Overo properly. These files can be downloaded from the database available at the Gumstix website, which is maintained up-to-date with the required files. The files are named: *MLO*, *u-boot*, *uImage* (these files are needed to boot the Overo) and a tar file (to deal with the OMAP3-console-image). We managed to use a USB Wi-Fi dongle, which supports Linux, to connect to the Overo as a plug-and-play for wireless communication.

Once the download is finished and all the updates are current, the next step is to create a bootable SD card. Two partitions will need to be created on the SD card for proper booting of the Overo Air COM. The basic files need to boot the Overo will be placed on the first partition, and the second partition will be for the Overo rootfs. First, we set up the parameters for the SD card. Next, we created the first partition for the card by setting its space capacity around 32MB. The second partition will cover the rest of the space on the SD card. After creating the partitions, the next step is to install the files needed to boot the Overo.

Three files are needed to properly initiate the Overo console session. The *MLO file* is needed for minimal configuration of the system's memory and the I/O pins. It is loaded into the OMAP3 processor's static RAM. The *u-boot file* is needed to load the u-boot prompt to communicate with the Overo and the *uImage file* is the Linux kernel for the entire system. The three files are copied to the SD card on the Overo using the *sudo cp* command. Once all three files are copied to the first partition of the SD card, all other necessary files and folders should be added to the second partition. The tar file that was mentioned earlier will need to be unzipped into the second partition of the SD card so that the desired Overo rootfs will be available.

After the SD card is formatted with two partitions with the necessary files, we are ready to set up the OE build environment. After creating a home directory folder, the OE metadata will need to be downloaded as well as *BitBake*. To retrieve, the *git* command is used.

First the required OE configuration files and scripts are put into the home directory to keep the environment ready for processing. These files will serve as the build system for the OE environment. OE requires an environment setup in order to function properly. This can be set up using the *bash* profile. After setting up the environment, we are ready to start our first initial basic build. We were able to build a basic kernel and non-gui root file system image with the BitBake command. The initial build will need some time to complete due to all the source code downloads and building of the cross compilation tools it has to do. After completing these tasks, the setup of the OE environment will be complete.

The final phase of the setup involves configuring the camera module. The issues to be dealt in this phase of the implementation are: (i) the SDA line, in the cable connecting the camera module to the Overo Air COM, should not have any short-circuit; (ii) the connectors need to be dust free and (iii) the use of a flex circuit cable for installing the module.

7. CONCLUSIONS AND FUTURE WORK

The high-level contribution of this paper is a research review of the different hardware and software components needed to successfully set up a wireless multimedia sensor network (WMSN) and a preliminary design for setting up a small-scale WMSN in an academic research environment. We also outline the procedure to implement the proposed WMSN system architecture. The Overo Air COM that we recommend for use with a WMSN has many applications, including monitoring a certain environment in a remote location to be capable of studying the conditions over a given time period. In order for the sensor node to fully operate, it must be programmed with the appropriate algorithms so that required configurations could be installed. Many additions can be made to the node also as needed for different applications. Instead of using batteries as a power supply, one may suggest to use solar power to make it more energy efficient. A GPS receiver would also be a respectable add-on; it could help locate other nodes that may be in the surrounding location. We plan to consider implementing these add-ons in the immediate future, before testing the WMSN design prototype and the implementation.

ACKNOWLEDGMENTS

This research is funded by the U.S. National Science Foundation through grant (CNS-0851646) entitled: "REU Site: Undergraduate Research Program in Wireless Ad hoc Networks and Sensor Networks," hosted by the Department of Computer Science at Jackson State University (JSU), MS, USA. The authors also acknowledge Dr. Loretta Moore, Dr. Scott Bridges, Mrs. Brenda Johnson and Ms. Ilin Dasari (all at JSU) for their services to this program conducted during Summer 2010.

REFERENCES

[1] I. F. Akyildiz, T. Melodia and K. R. Chowdhury, "Wireless Multimedia Sensor Networks: Applications and Testbeds," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1588-1605, 2008.

[2] I. Downes, L. B. Rad and H. Aghajan, "Development of a Mote for Wireless Image Sensor Networks," *Proceedings of Cognitive Systems and Interactive Sensors*, Paris, March 2006.

[3] Gumstix, "Overo Air COM," Available at http://www.gumstix.com/store/catalog/product_info.php?products_id=226, last accessed: August 22, 2010.

[4] Gumstix, "Summit expansion board," Available at http://www.gumstix.com/store/catalog/product_info.php?products_id=215, last accessed: August 22, 2010.

[5] e-con Systems, "e-CAM32_OMAP_GSTIX," Available at http://www.e-consystems.com/omapovero.asp, last accessed: August 22, 2010

[6] AWUS036H, "Wireless USB adapter," Datasheet available at http://site.microcom.us/AWUS036H.pdf, last accessed: August 22, 2010

[7] McHugh, Sean. "Lens Quality: MTF, Resolution & Contrast," Available at http://www.cambridgeincolour.com/tutorials/lens-quality-mtf-resolution.htm, last accessed: August 22, 2010.

[8] CarltonBale.com, "1080p Does Matter – Here's When (Screen Size vs. Viewing Distance vs. Resolution)," Available at http://carltonbale.com/1080p-does-matter, last accessed: August 22, 2010.

[9] TheBestPlasmaTV.com, "Plasma TV Size & Optimal Viewing Distance," Available at http://www.thebestplasmatv.com/guides/size-optimal-viewing-distance/, last accessed: August 22, 2010.

[10] OMAP 3503 Application Processor, Datasheet available at http://focus.ti.com/lit/ds/symlink/omap3503.pdf, last accessed: August 22, 2010

[11] W2CBW003B Wi-Fi Module, Datasheet available at http://www.wi2wi.com/products/datasheets/W2CBW003_PB%20rev1.2.pdf, last accessed: August 22, 2010.

[12] TPS65950 Power Management Module, Datasheet available at http://focus.ti.com/lit/ds/symlink/tps65950.pdf, last accessed: August 22, 2010.