2008

WIRELESS INTELLIGENT STRUCTURAL HEALTH MONITORING SYSTEM

Belle Ashwin
Virginia Commonwealth University

WIRELESS INTELLIGENT STRUCTURAL HEALTH MONITORING SYSTEM

A thesis submitted in partial fulfillment of the requirements for the degree of Master Computer Science at Virginia Commonwealth University.

by

ASHWIN BELLE
Bachelor of Engineering, Dayananda Sagar College of Engineering, India.

Director: DR JU WANG
ASSISTANT PROFESSOR, DEPARTMENT OF COMPUTER SCIENCE

Virginia Commonwealth University
Richmond, Virginia
December, 2008

Acknowledgement

First and foremost, I would like to thank Dr. Ju Wang for being such a great advisor and a great friend. I would also like to thank Dr. Ramana M. Pidaparti for having given me several great opportunities to work alongside him on his projects. I would like to give my special thanks to Dr. James E. Ames for providing me with all the resources throughout my graduate studies. In addition, I would like to thank the faculty of the Computer Science Department for facilitating the excellent academic experience that I have had here at Virginia Commonwealth University. I would also like to extend my thanks to National Science Foundation for having facilitated the project with their support and funding for this project.

I would also like to thank my family for their everlasting support. Finally I would like to thank Paras M Gandhi for his supportive, friendly nature and expert help with the Linux system.

Table of Contents

List of Figures

Abstract

WIRELESS INTELLIGENT STRUCTURAL HEALTH MONITORING SYSTEM

By Ashwin Belle

A Thesis submitted in partial fulfillment of the requirements for the degree of Master Computer Science at Virginia Commonwealth University.

Virginia Commonwealth University, 2008

Major Director: Dr Ju Wang, Assistant Professor, Department of Computer Science

Metal structures are susceptible to various types of damages, including corrosion, stress damage, pillowing deformation, cracks etc. These kinds of damages in the metal structures occur mainly due to operational conditions and exposure to the environment. Our research involves a portable integrated wireless sensor system with video camera and ultrasound capabilities which is being developed to investigate corrosion damage on real structures in real time. This system uses images of the metal surfaces, which are captured from an integrated wireless sensor and then quantified and analyzed using computational intelligence. The quantification of the obtained images is done with specialized component analysis software which enhances and performs wavelet transforms on the received images. Through this quantized analysis of the images we can detect and isolate regions of degradation on the metal surface. We believe that the final developed system will allow us

to detect damage in metallic structures in its early stages, thereby ensuring proper safety and maintenance of its structural health.

This system will further be targeted towards medical applications with capabilities of remote health monitoring. The initial target areas being bone structure and cancer detection and analysis. Applying such a wireless data capture system in these areas will reveal a broad spectrum of the usage of such an application system.

1. Introduction

Damage and degradation of metal is a major mechanism affecting the structural integrity, public safety and economy of industries around the world. Metal structures are susceptible to various types of such damages, including corrosion, stress damage pillowing deformation, cracks etc. These kinds of damages in the metal structures of the vehicles occur mainly due to its operational conditions and exposure to the environment. Our research involves a portable integrated wireless sensor system with optical sensor and ultrasound capabilities which is being developed to investigate corrosion damage on real structures in real time.

Several Nondestructive evaluation (NDE) systems (eddy current inspection systems, enhanced visual inspection system or stereographic system, acoustic emission, and others) have been used to detect the cracks/damage in aging structures and materials[1-4]. Nondestructive evaluation procedures involve establishing correlations between measured properties and quantitative information about anomalies. Also, because of damage tolerance requirements, detecting and quantifying even the smallest damages is a critical aspect of NDE systems. Manual identification and quantification of damages based on NDE images is a tedious and subjective task. There have also been several studies dealing with structural health monitoring using fiber optic sensors[5] and wavelet methods[6]. To complement existing methods of structural health assessment, computational intelligence methods can also be used. This system that we are developing uses images of the metal surfaces which act as two dimensional signals and can be more informative than one dimensional data for analysis and detection of corrosion. The image can be derived from various sensors such as ‘Eddy current’, ‘Ultrasound’ and “Optical Sensor”. Such NDE procedures involve establishing correlation between measured properties and quantitative information about anomalies.

This research is based on the development of an embedded handheld wireless computing system which is capable of capturing images with an image sensor and relaying the captured images wirelessly to remote locations for analysis[7, 8]. The device is equipped with a high resolution image sensor, which can dynamically capture JPEG image at different resolutions. The device is also being designed such that various other forms of sensors can be attached onto it and used to capture data for analysis. The device contains a wireless interfaces ‘W-CDMA’ long range interface to transport the data directly to our image server. Once the captured images are received by a server station wirelessly, these images are sent through specialized component analysis software. This software automatically detects and quantifies corrosion by enhancing and performing wavelet transforms on the received images. A principle component analysis is done on the data to reveal aspects such as trends, breakpoints, discontinuity, and similarities. Based on the resulting quantified values a decision can be made upon the necessary actions to be taken on the surface subject. Finally after analysis of the image some key *Analyzed Information* is relayed to the technician on the aircraft about the analyzed metal surface. This is done with the W-CDMA wireless transmission from the server station back to the handheld computing device. A user-interface has been developed on the handheld device to show raw images, analyzed data and provide guidance/instructions pertaining to the required actions.

The developed system allows us to detect damages in the metal structures in its early stages thereby allowing us to ensure proper safety and structural health maintenance of engineering structures such as pipelines, nuclear reactors, mines etc.

The developed system may also find applications in the healthcare area. The system can be further developed into finding applications in certain key areas of healthcare. An example application of this device with healthcare would be using the portable handheld device to capture images if possible bone fracture victims in remote areas and then, relaying them wirelessly in real time to hospitals of doctors who can diagnose the problem without having to be present with the victim. Thereby allowing physicians to make life saving decisions for the patients in real time without wasting valuable time.

2. The system ideology

The following example is designed to further explain the working model of the whole system based on the design developed for the experimental purposes. The example is illustrated based on how the system is designed to be used from start to end in a typical usage scenario.

Let us consider that there is an aircraft which is now stationed in a remote part of the world. The technicians on the flight believe that there might be some damage on the metal structure of the plane due to its age. Now consider that the technician on board is equipped with the handheld device which is being developed as a part of this project. The technician can directly go to the region on the plane where the metal surface seems to have some wear and tear or some abnormalities. He can then capture images using the handheld unit of the metal surface which needs analysis and decision in terms of what has to be done with that area of the metal structure. These images can then be wirelessly transferred by the handheld unit to the server which may be present in the headquarters of the airline company half way around the world.

These images are then received by the server machine at the headquarters where qualified engineers trained to analyze such metal corrosion and structure defect are present. The received images automatically get store in a repository with the details of when it was taken and where it originated from. Based on this the images received are then sent through the Component analysis software present on the server machine. The component analysis software performs detailed image processing on the acquired image, thereby being able to extract and segment those regions of the metal surface which is suspected to be corroded. The software also gives a quantitative value of how much corrosion and material loss has taken place on each particular area of corrosion. Based on these values the engineers analyze the quantitative data and come to a decision based on what further action is to be done on the corroded region.

The decision here can be based on various aspects which the engineers understand through their accumulated knowledge of the system and the structure from which they received the images. Based on what decision they would then relay back the instruction to the technician on board the aircraft to work on the metal surface accordingly. An example of such a decision could be that the engineers after analyzing the data realize that there is too much damage on the metal surface thereby requiring a complete change of that metal sheet so as to not compromise the integrity of the metal structure any further.

The feedback from the engineers from the headquarters would be presented to the onboard technician on the handheld unit in a very easy to understand and presentable manner. This kind of a system allows for a technician who is not as qualified to be able to actually work and repair structure based on expert decisions which are wirelessly relayed to him from remote locations. Thus, allowing precise decision making skill and real-time problem solving to be implemented on critical situations.

3. Embedded System Interface - the first design

For the development of a handheld device module of the proposed system, we had to first recognize the needs of the different applications the system will be used for. The idea was to make the handheld unit as simple and versatile as possible, while ensuring reliability and efficiency. One of the key aspects of a handheld unit such as this would always be its portability. So keeping this in mind, the development of the device was targeted towards a simple wireless computational device which was specifically designed to capture images with the necessary sensors and transfer them wirelessly to the server.

3.1 The First Design Setup

The most important part of the device would be its processor and its computational capabilities. During the first stages of the development the processor chosen for this was the Atmel Mega1281. The ATmega1281 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture[15]. ATmega1281 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications. By executing powerful instructions in a single clock cycle, the ATmega1281 achieves throughputs approaching 1 MIPS per MHz allowing the system designed to optimize power consumption versus processing speed.

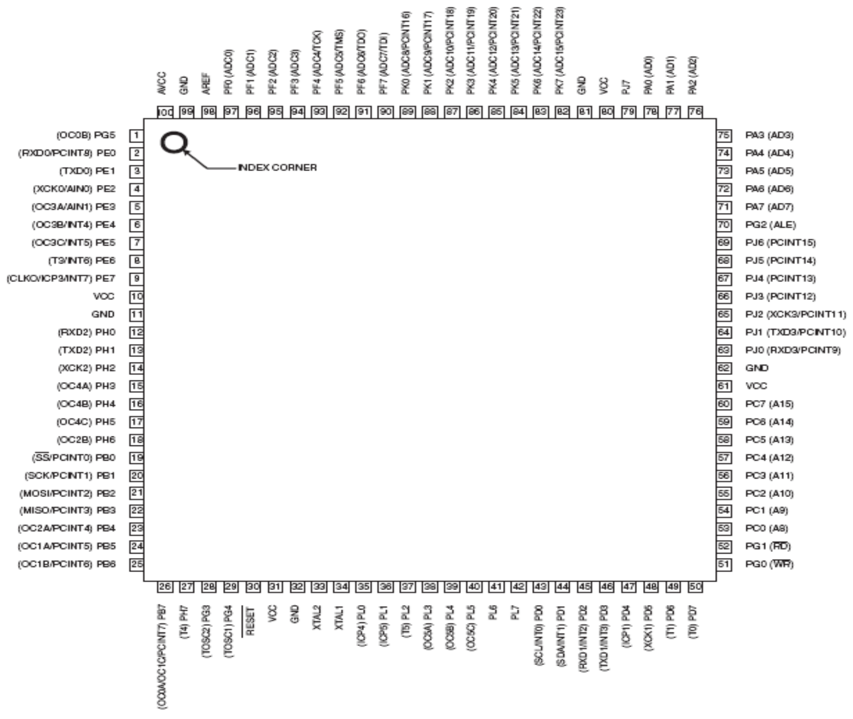
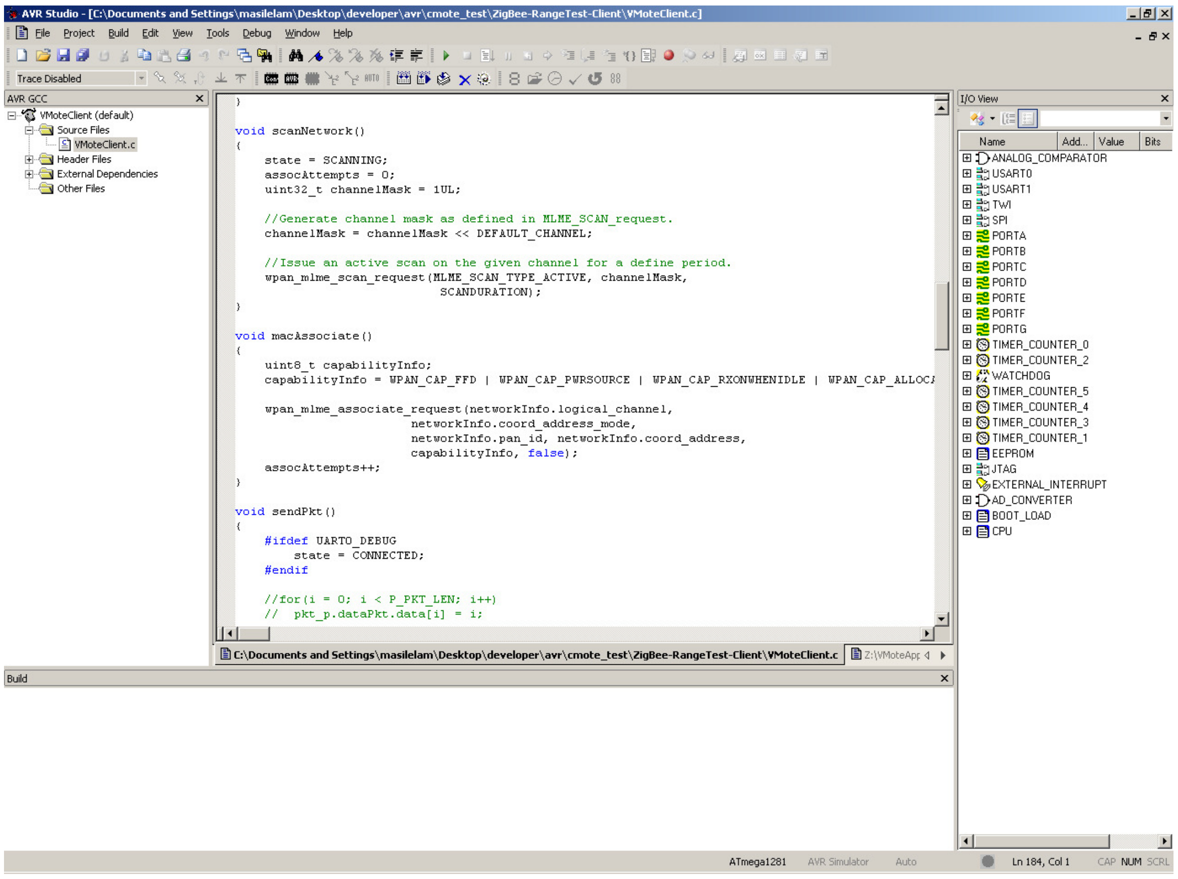


Figure 1: ATMega1281 pin diagram

Being a low power consuming microprocessor along with the AVR core having a rich instruction set and 32 general purpose working registers, it was found ideal to be used for this application. In order to program the wireless sensors, we use AVR Studio. This is a programming platform provided by Atmel, the manufacturer of the microcontroller used in the wireless sensors.



```
}

void scanNetwork()
{
    state = SCANNING;
    assocAttempts = 0;
    uint32_t channelMask = 1UL;

    //Generate channel mask as defined in MLME_SCAN_request.
    channelMask = channelMask << DEFAULT_CHANNEL;

    //Issue an active scan on the given channel for a define period.
    wpan_mlme_scan_request(MLME_SCAN_TYPE_ACTIVE, channelMask,
                            SCANDURATION);
}

void macAssociate()
{
    uint8_t capabilityInfo;
    capabilityInfo = WPAN_CAP_FFD | WPAN_CAP_PWRSOURCE | WPAN_CAP_RXONWHENIDLE | WPAN_CAP_ALLOCA

    wpan_mlme_associate_request(networkInfo.logical_channel,
                        networkInfo.coord_address_mode,
                        networkInfo.pan_id, networkInfo.coord_address,
                        capabilityInfo, false);
    assocAttempts++;
}

void sendPkt()
{
    #ifdef UART0_DEBUG
        state = CONNECTED;
    #endif

    //for(i = 0; i < P_PKT_LEN; i++)
    //  pkt_p.dataPkt.data[i] = i;
```

C:\Documents and Settings\masilelam\Desktop\developer\avr\cmote_test\ZigBee-RangeTest-Client\VMoteClient.c

Figure 2: AVR Studio Programming enviornment

The AVR programming is done using Embedded C programming language. It is similar to C but the programs are specifically designed for target architecture. In this case the architecture being the ATmega1281. Once the program is developed the program is then converted to a hex format which is then fed into the ATmega microcontroller. The code that was developed for the application was designed to talk to the camera module which was integrated to the system and capture images. After having captured the image the picture would then be transmitted wirelessly using a popular wireless protocol to a remote server for analysis.

To capture images the system has been given a broad spectrum design such that various kinds of sensors can be employed in the system to capture the data that can be analyzed. For the initial developmental stage however the optic sensor was as the method for experimental data capture. The sensor selected for the integration into the handheld unit was the C328-7640 JPEG Compression VGA Camera Module.

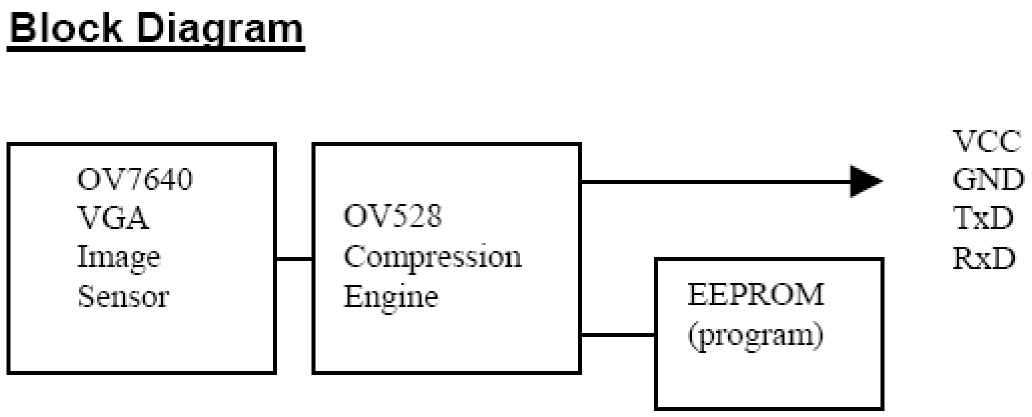


Figure 3: Block Diagram of the Camera Module

The C328 JPEG compression module performs as a video camera or a JPEG compressed still camera and can be attached to a wireless or PDA host. Users can send out a snapshot command from the host in order to capture a full resolution single-frame still picture. The picture is then compressed by the JPEG engine and transferred to the host. The C328 camera module is a low power, low cost solution for integrating an optic sensor into the system. Its built in compression engine enabled us to get either raw data or JPEG compressed data as and when required. Furthermore its small size and good image quality made it an ideal choice as an optical data capture agent

For initial experimental setup the wireless data transmission was done using ZigBee™/IEEE 802.15.4- Transceiver. Zigbee is IEEE 802.15.4 standard for wireless data transmission. It can be operated with very low-power consumption. It operates in the 2.4 GHz frequency spectrum which is part of the ISM [Industrial, Scientific, and Medical], thereby allowing us to freely use this standard without the need for any licensing for operating in this frequency range. It has several different channels for choice of operation. Transmission distance is expected to range from 10 to 75 meters, depending on power output and environmental characteristics.

The first design implementation of this Portable Integrated Wireless Sensor System operated in several discrete but interdependent steps. The stage of capturing the images and replaying it to remote server wirelessly was integrated using the above mentioned device interfaces. As mentioned earlier our chosen method for Nondestructive Evaluation is based on images captured of the metal surfaces under observation. This capture stage consists of a camera, micro-controller and a wireless transmitter of data. The required images can be obtained using various sensors such as ‘Eddy current’, ‘Ultrasound’, ‘Video Camera’, ‘Eddy current’, ‘X-ray’ etc. Our experimental system is primarily being developed with a video camera sensor to capture direct images of the surfaces to be investigated. This same concept can be extended in the system to include other sensors mentioned as well.

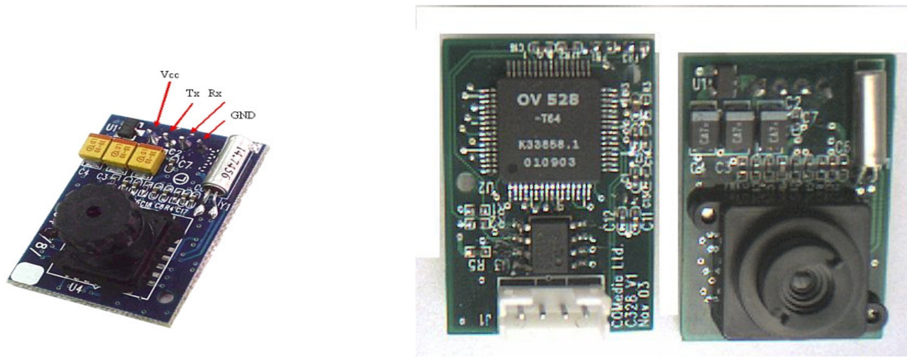


Figure 4: C328 camera module

The C328R is a VGA camera module. It functions as a JPEG compressed still camera and can be attached to a wireless or PDA host. The camera can send out a snapshot command from the host in order to capture a full resolution single-frame still picture. The picture is then compressed by the JPEG engine and transferred to the host through a serial port. The camera is capable of capturing images in various resolution formats ranging from 80x64 pixels to 640x480 pixels[14]. These resolutions can be preset or changed based on operational modes. There are three different capture modes present on the camera, Snapshot picture, Preview (video streaming) and JPEG Preview Picture. The pictures obtained can either be compresses on uncompressed.

The camera module does not operate by itself as required. To operate the camera module a program was developed for the microcontroller chip. This microcontroller chip initiates the camera and interacts with it with specific commands. The sequence of commands is based on a preset binary language code which the camera has been trained to respond to.

- **Start**
    - Power up the camera
    - Send **SYNC** message. (Synchronize) x 30
    - Receive **ACK** from camera (Acknowledgment).
    - Receive **SYNC** from camera (Synchronize feedback)
    - Send Ack. (Acknowledge feedback).

- Send Initialize packet.
    - Receive **ACK** from camera
- Send Set Package Size Packet.
    - Receive **ACK** from camera
- Send Get Picture packet
    - Receive **ACK** from camera.
    - Receive the **JPEG** compressed picture packets.

When an image is to be captured the microcontroller chip does a capture image routine with the camera. As shown in the step wise operation procedure the microcontroller first powers up the camera and then it initiates the camera by sending a Synchronization command (SYNC). The camera responds by sending back an Acknowledge command. Then the microcontroller sends the camera specific commands which instruct the camera about the type of picture, picture quality and size. Then the camera sends the Get Picture command. Upon receiving this Get Picture command, the camera takes a still shot capture through the sensor and records the data in its memory. The recorded image is then compressed into the JPEG (Joint Photographic Experts Group) format by the camera before relaying the image data to the microcontroller.

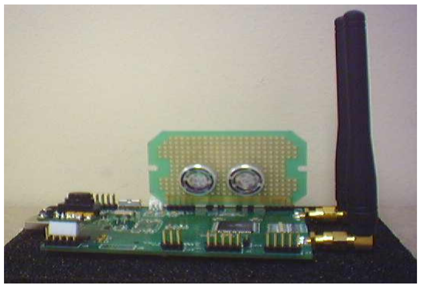


Figure 5: Wireless Zigbee Module (Transreceiver).

The microcontroller at this point has the acquired image and this has to be sent back to the server station where the image is to be analyzed and quantized. The data transmission here has been designed such that the server station can be in a completely remote location. For the wireless transmission we have developed a Dual ZigBee wireless transceivers board. ZigBee is the name of a specification for a suite of high level communication protocols using small, low-power digital radios based on the IEEE 802.15.4 standard for wireless personal area networks (WPANs). To implement the Zigbee protocol, we have built the wireless module with an Atmel AT86RF230[9]. The AT86RF230 is a low-power 2.4 GHz transceiver specially designed for low cost ZigBee/IEEE802.15.4 applications. This Zibee is an ideal platform for the basic implementation of our system since it has a relatively reliable wireless communication stack structure. Also this allows us to transmit the data through the wireless module and manage it with a buffer management system as well as stores it in a database for future use and record maintenance. Hence the wireless integrated board and camera module together form the components of the first stage of the system. The images captured at this stage are then relayed to the next stage, the Analysis stage.

3.2 Design Faults of the First Design

As the experiments were performed under this initial setup, the various pitfalls of this system were soon apparent. One of the biggest concerns that arose during the development of this handheld unit was that of the inefficient operation of the camera module with the ATmega1281 processor. There were a lot of issues experienced with the proper integration of the camera sensor. The problem was the instability of the sensors operation. It was found extremely troublesome to sometimes get the camera operational and in sync with the processor.

Another problem with this system was the wireless aspects. Although Zigbee seemed like a very easy and efficient choice, its range and reliability of service was an obvious hindrance for the complete mobility of the handheld system in its true sense.
So considering the various design faults and implementation issues that were encountered in this first experimental setup, a new setup was designed for this system.

4 Embedded System Interface - the Second design

Due to some key features being compromised in the first design of the embedded handheld device, it was then decided to develop a newer revised version of the handheld unit. The focus again was on the most important aspects that the device was being designed for. Some of the aspects that were more emphasized on with the revision of the handheld unit were the wireless technology, the processing capabilities and the sensor module.

The plan was to develop the device and make it as portable as possible with exceptional computing power for such a small device, at the same time keeping the usability of the device to as elementary as possible so that anybody with just little training should be able to operate it.

4.1 Revised Design Setup

For the new experiment setup a few essential changes were made to encompass the performance requirements and integration of the system. The first big change over the old system was the usage of the Gumstix micro computer for the processing needs of the system as compared to the ARV Atmel chipset. For the camera sensor a new and improved version of a Logitech PC webcam was used which was integrated into the working of the Gumstix processor. Finally with the wireless front the decision was made to use the WCDMA commercial unit for reliable and highly portable data transfer.

With our first design setup a lot had been learnt with its inadequacies when it comes to the essential needs of a portable handheld unit such as this. It was very clear that processing power is a key requirement to have a powerful system. Moreover, the processor also needs to be versatile since easy interfacing would be required to run various sensor modules on it. Another clear aspect was that for experiment and demonstration purposes also, we would require a better camera sensor which would first and foremost be more reliable than the older one and secondly would give us a better quality data thereby making the system a better benchmark for quantitative analysis.

Finally it was certain that to actually have a truly portable system a better and more reliable wireless protocol would be required in the system. Thus, allowing the user to fully make use of its portability and usability.

4.2. Processor Module

Gumstix motherboards drive a fully function, gum stick size Linux computers. Each gumstix motherboard is packed with the power and performance of much larger SBCs (Single Board Computers), making gumstix motherboards perfect as the brains of network management appliances, handheld devices, industrial monitors, robotics and much more.

Gumstix motherboards come in two active product lines; the smaller than a gum stick, Overo™ series based on Texas Instrument OMAP® 3503 and gum stick size verdex pro using Marvell® PXA270 with XScale™. For the development of this project the Verdex pro with the Marvell chipset seemed ideal and has hence been chosen as the primary processing module. Every motherboard comes pre-flashed with Linux 2.6 and Linux OpenEmbedded and has been designed to run standalone, or with additional function available via optional expansion boards.

These motherboards are ARM®-based platform which run on low cost Linux system images. By using Linux the design could also take advantage of the 256MB low power DDR RAM, 256MB NAND flash, an on-board microSD adapter, a 24-pin Hirose FH26-27S connector for camera control signals and two (2) x 70-pin AVX 5602-14 connectors for a wide range of functional options in expansion board design.

Verdex pro motherboard offers on-board storage with microSD and flexibility by way of adding two expansion boards: one on each side of the motherboard.

The Gumstix processor is the Verdex pro XLP6. The XLP6 is a fast motherboard in the PXA270-driven verdex pro product line. With the on-board memory slot and optional 10/100 and wifi connectivity, this verdex pro motherboard becomes the heart inside demanding mobile, remote and engineering devices.

Figure 6: Verdex pro XL6P processor module

The processor has a 600Mhz Marvell® PXA270 with XScale™ as the CPU. For the higher memory requirement of this system the processor has been expanded with a 2GB micro SD Card. The Linux Image has been built onto the SD card for easy booting and expansion. The implemented Linux kernel is the 2.6.21 version. The device has been setup such that the booting can directly take place from the SD card.

4.3. Expansion boards

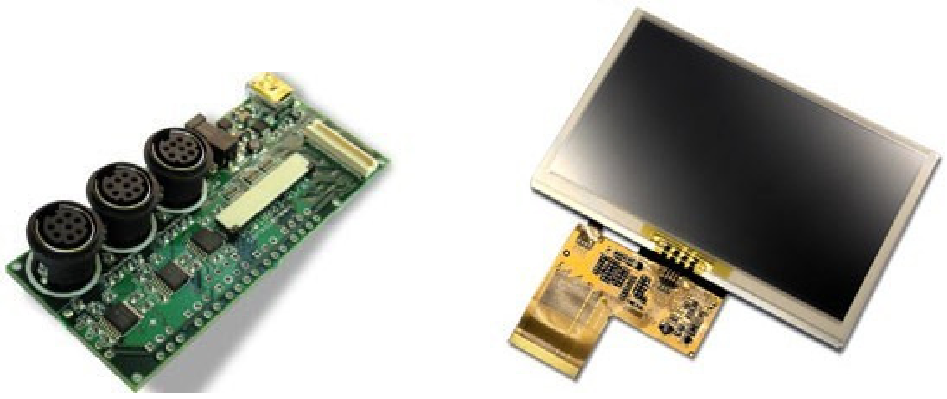


Figure 7: LCD 16-vx expansion module

The console LCD16-vx is an expansion board for the touch resistive LCD color panel. This board allows us to integrate the Samsung LCD panel with resistive touch screen overlay. This touch screen adds a level of sophistication as well as usability to the overall system. The console LCD16-vx expansion board comes along with three RS-232 ports on miniDIN8 connectors. Thus, allowing several I/O interaction capabilities. It also hosts a USB mini-B connector with USB Host signals. The boards come LCD-ready along with a built in Touch screen controller. The board also provides GPIO lines and an I2C port, allowing us to connect various kinds of peripheral devices to the system.

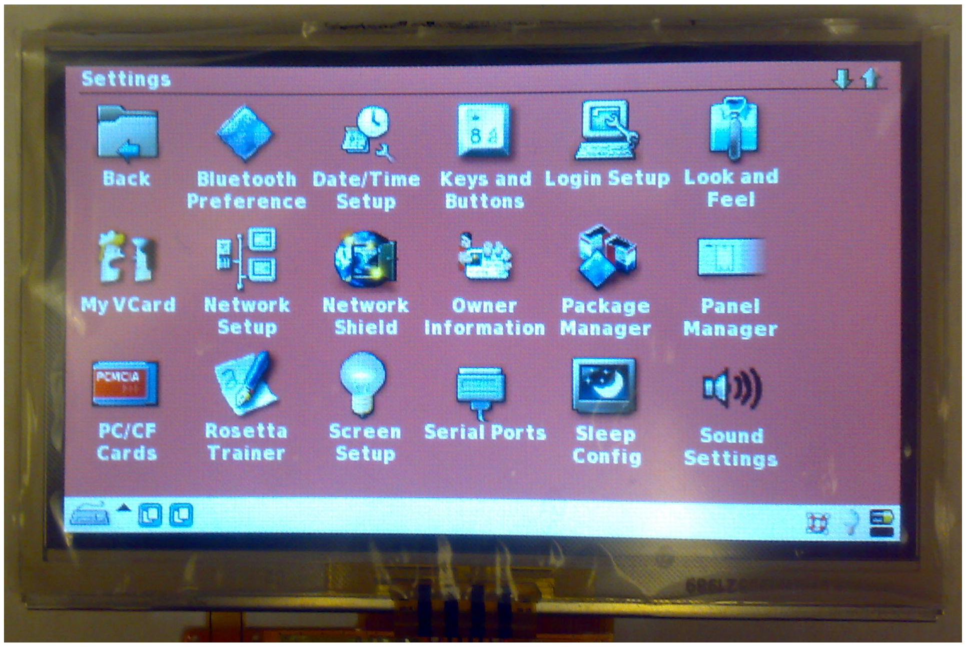


Figure 8: samsung LCD touch-screen panel

The Samsung LCD is a 4.3" diagonal screen. 480x272 pixels, max 8 bits each of R,G,B with resistive touch screen overlay. The touch screen allows a very high level of interaction between the user and the system.

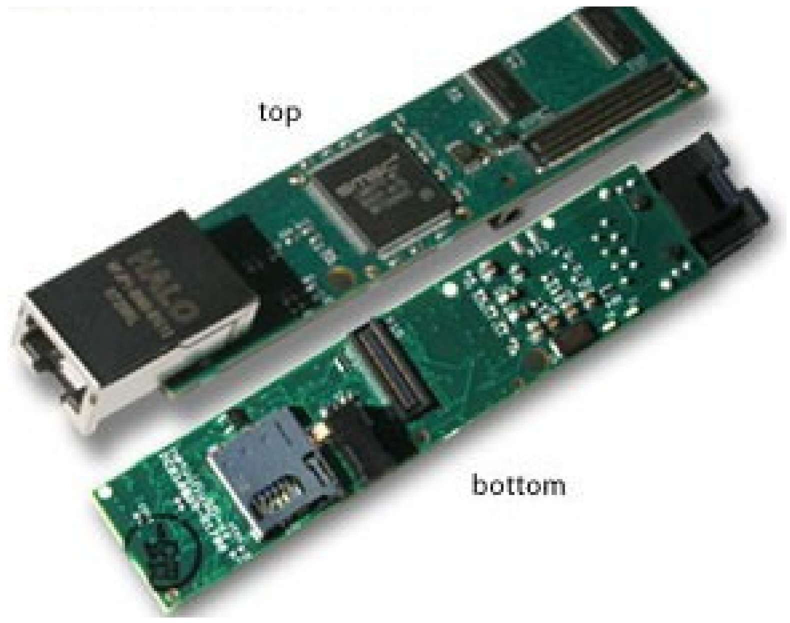


Figure 9: NetmicroSD-vx expansion board

The NetmicroSD-vx expansion board facilitates the system with networking interfaces such as LAN and optional Wifi. Also another important aspect of the expansion board is the Mirco SD card memory expander. This allows us to simply increase the memory capacity of the board from 128MB to up to 2GB.

4.4. Installing the Linux image in the Gumstix

It is to be noted that the gumstix is of the ARM architecture. So each operation and the Linux image itself must be cross compiled onto the arm architecture before being flashed onto the gumstix board. For an Ubuntu 7.10 Linux was first setup on a Desktop in our lab. This desktop is used as the server for the cross compilation and the image processing.

Applications
Places
System
Sunday April 29th, 2007 3:59pm
ubuntu
Desktop 2
Mozilla Firefox
connected to : T-mobile (wireless)
1
2
3
4

Figure 10: Ubuntu Linux Operating System

For the actual cross compilation platform the gumstix supports the OpenEmbedded toolchain. The OpenEmbedded Project (OE for short) is a software framework to create Linux distributions aimed for embedded devices. Primarily, the project maintains and develops a collection of BitBake recipes, similar to Gentoo's ebuilds. The bitbakes consist of the source URL of the package, dependencies and compile or install options. During the build process they are used to track dependencies, cross-compile the package and pack it up, suitable to be installed on the target device. It's also possible to create complete images, consisting of root file system and kernel.

As part of the build process Gumstix OE downloads source code tarballs for the linux kernel and other software packages. This step sets up a global system cache for these tarballs so that they only need to be downloaded once. Setting up a global cache may seem like a lot of trouble for a single user system, but it is often quite useful to set up an autobuilder that runs as a cron job under a separate user account.

Although the gumstix has a prebuilt image ready to go out of the box, a new image had to be built for the more sophisticated needs of the system under development. Hence the newer Linux image was built with the target application in mind. The newer image has the GUI (Graphics User Interfacing) interfacing rather than just a command line interaction setup. The newer image also supports the touch screen feedback thereby enhancing the user interaction with ease of use while making this happen in a more presentable manner.

Due to the enhancements in the new image, the actual size of the image was found to be considerably larger than the standard pre built image. Thus a MicroSD card of size 2GB was installed to expand the memory space so as to accommodate the larger sized Linux image.

The MicroSd Card is of size 2GB. It had to be partitioned into two separate drives for proper separation of the file system and the kernel image. The first partition is a 64MB partition. Its file system is formatted to the Fat16 variant. The Linux boot script and the Linux boot image are stored in this first partition. As the system boots the Built in Uboot of the gumstix module recognizes the image on the MSD card and automatically loads the boot script to load the kernel.

The second partition occupies the rest of the storage space and is formatted to the EXT2 type file system. This partition will host the root file system of the Linux OS.

The whole assembly of the gumstix including the Linux image and the LCD screen looks as follows. This is the picture of the actually computer on module that we care using for the system.

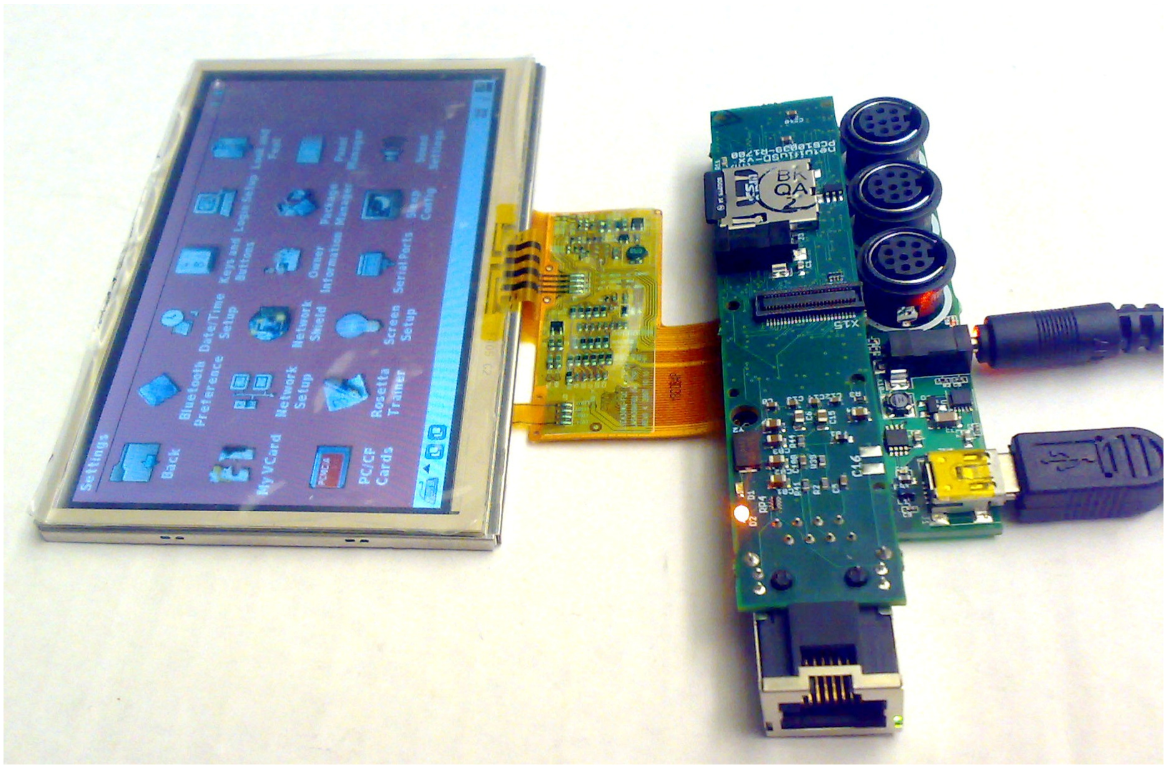


Figure 11: Complete Gumstix Setup

4.5. **Camera Module Installation**

For the Image capturing the Logitech QuickCam Pro 9000 was chosen. This webcam is a high resolution webcam with a 2 mega pixel resolution. The QuickCam Pro 9000 is built with a Carl Zeiss® lens, with the Ultra-high resolution 2-megapixel sensor using the RightLight™2 Technology. The camera can capture images with a frame rate of up to 30 frames per second. It's a well designed highly compact image capture device.

Figure 12: Logitech QuickCam Pro 9000

Although it has been primarily designed to work on the windows PC platform, there are certain open source drivers available that will integrate this webcam to work with a linux platform. But in this case not only did the camera have to work with the linux image that had been built specifically for gumstix, it also had to be integrated onto the arm architecture. Hence using the BitBake OpenEmbedded cross compiler, the open source driver for this called the UVC driver was crossed compiled to work on the gumstix platform with the new image.

For a webcam such as this to work on a gumstix Linux package certain special drivers and library functions are essential. The one that was found needed here was the V4L framework or also known as the Video4Linux framework. Video4Linux or V4L is a video capture application programming interface for Linux. Several USB webcams, TV tuners, and other devices are supported. Video4Linux is closely integrated with the Linux kernel.

Upon installing the needed framework and drivers onto the Linux platform we had the Logitech webcam operational on the gumstix. Thus allowing us to capture images and store them on the local memory.

4.6. Wireless Module

The previous design model had ZigBee as its wireless protocol. For experimental purposes the Zigbee was chosen as a good wireless protocol due to its ease of operation and effectiveness. But due to its shortcoming such as its short range and lesser reliability, it was decided that a different wireless protocol would be necessary to be implemented for use in this system.

So, the CDMA protocol was chosen. In this newer design frame we have incorporated a commercially available wireless data card by Sprint. This wireless data card functions on the CDMA technology and can be used in all places where the CDMA network reception is available.

Figure 13: Seirra Wireless AirCard 595U by Sprint

The AirCard® 595U USB Wireless modem is a stylish, compact and highly reliable. the AirCard 595U USB modem can either plug directly into any USB port, or can be connected via the included docking cradle. It was designed to be primarily compatible with Microsoft Windows and Mac OS X operating systems, but there are some support pages available for it to be run on the Linux OS.

The AirCard supports wireless data transfer at max speeds of 3.1 Mbs downlink and 1.8 Mbps uplink, It had the EV-DO Rev A networks High performance internal antenna for measurable improvements in data speed, signal acquisition and retention. It supports Dual-band, 800 and 1900 MHz, with built-in GPS antenna[16].

The AirCard support for Linux users is facilitated using a sierra.c driver(v.1.0.6) which is compatible with Linux kernel version 2.6.17 or higher. To make it run on the gumstix it was required to have certain PPPD scripts running. PPPD is the Point-to-Point Protocol daemon which is used to manage network connections between two nodes on Unix-like operating systems.

4.7. Packaging of the components into a single handheld unit

To encompass the several different components of the whole system as a one single handheld unit a packaging prototype was designed and built for the system.

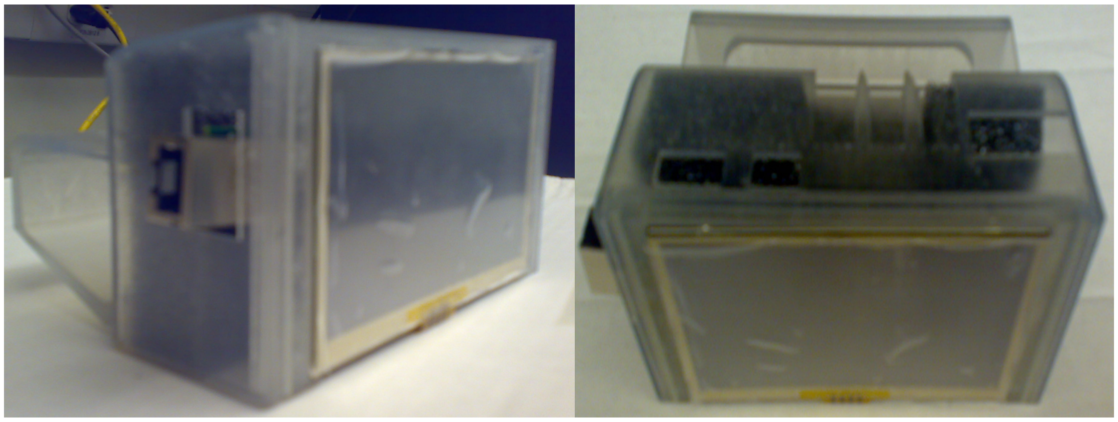


Figure 14: Resin casing for Gumstix

The above figure shows the simple handheld unit casing that was developed to contain the several different components of the system. The casing allows the handheld unit to be used as a standalone portable unit with ease of carrying and data porting. The body structure of the casing has been developed in such a way that the attachment of the peripheral devices and sensors can be done through the various input output cable openings that have been provided. Also for easy assembly and disassembly the top of the casing has a slid cover upon which the LCD screen resides.

The casing was developed in VCU itself, with the help of the Mechanical Department. The modeling for the casing was done in a modeling software called ‘Solid Works’. To put the casing model into use we developed a model of it with rapid prototyping techniques. The working prototype was developed using solidification of liquid resin by laser. To do this we feed the prototyping computer with the software copy of the model designed. The model is then prepared by an automated process which produces it layer by layer by solidifying liquid resin at each stage.

The dimensions are based on the size of the Gumstix computer on module. It took roughly 16 hours to produce such a rapid prototype of a working model of the handheld unit.

5 Software Design and Development

Once the handheld device has captured the data through its sensor it then relays this information to the server system wirelessly. The computational intelligence for quantification and analysis of the NDE (Non Destructive Evaluation) images is done with a java based Program code specifically designed for image analysis and corrosion detection.

At the heart of the analysis structure is the Wavelength transform technique. In the past, wavelet transforms have been used for structural health monitoring primarily based on signal processing related to detecting damage in structures[10]. In our study, we have incorporated it into out computational analysis stage to process the obtained NDE images from the Capture stage.

Usually, images obtained through conventional NDE methods are not suitable for direct identification and quantification of damaged regions in aging aircraft structures due to variability of various NDE techniques. The Analysis stage begins with the Zigbee based wireless interface receiving the NDE images from the wireless camera module in the first stage. The received image is then fed as the input to the analysis software running in the server station. To increase efficiency of the wavelength transform analysis, the image is first enhanced. The enhancement is a software based pixel enhancement method. The wavelength transform method is then applied onto the enhanced image[11]. Wavelet features have been used for texture segmentation and for estimating the material loss due to corrosion since wavelet analysis is capable of revealing aspects of data that other signal analysis techniques lack, such as trends, breakdown points, discontinuities in higher order derivatives, and self-similarity[12]. After applying the image wavelength transform we still have to identify those portions of the surface image that are suffering from corrosion. For this purpose, we apply identification and segmentation routine which is based on the quantitative values obtained from the wavelength transform. It separates out only the corroded and damaged section of the target surface in the image. The quantitative analysis also associates each of the identified segments with an estimation of the percentage of metal loss at the damaged region.

Conventional Non-Destructive Evaluation techniques using signals from ultrasound and eddy current methods contribute to poor signal-to-noise ratio and interfere with the damage signals. Further, it is not possible to determine parameters such as thickness or the morphological properties of the damage. This poses a problem for accurate characterization and quantification of damage due to corrosion and widespread fatigue in aging metal structures.

Hence we to address the analysis of one type of damage that appears on most structures, that is corrosion. Corrosion identification and analysis can be considered as texture analysis process. It is particularly famous in the field on artificial intelligence. Several methods are being researched and developed in the field of artificial intelligence. The computational intelligence that can be applied to such texture analysis methods allow us to apply this in automated inspection problems in various fields such as corrosion detection, medical applications etc. This also incorporates a sense of remote based sensing and analysis[1].

The java program that is being developed is designed with respect to a broad region of application possibilities that this system can be subjected to. But for the experimentation and demonstration purposes the completed entities are now just with the corrosion detection aspects. The program is designed in a user friendly manner with a pleasing GUI (Graphics User Interface) and a very interactive interface.

The program once compiled and run will first come up with the home page. The Home page of the software looks as follows;

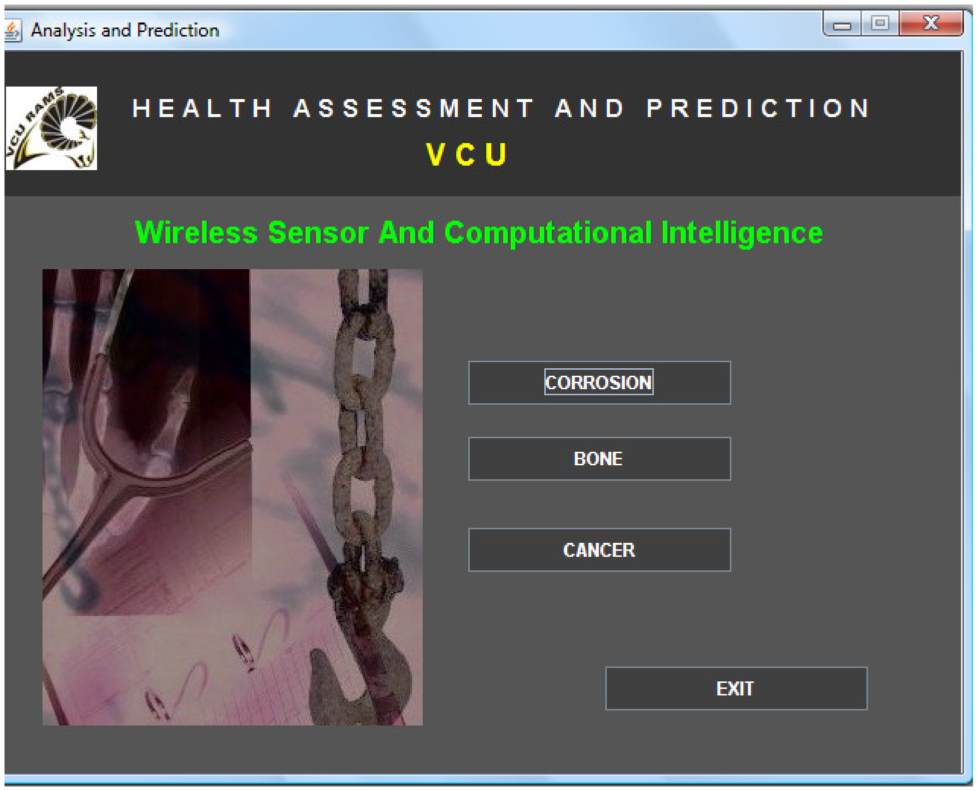

Analysis and Prediction
HEALTH ASSESSMENT AND PREDICTION
VCU
Wireless Sensor And Computational Intelligence
CORROSION
BONE
CANCER
EXIT

Figure 15: Home Page

Here we have a choice of which functional aspect we choose to work with. The three available choices at this point of time are corrosion, bone and cancer. The corrosion button leads you to the options of analyzing and prediction metal corrosion with images obtained from the remote handheld sensor. The Bone button is designed for medical applications pertaining towards bone structure analysis which is something that will be developed in the next stages of the system development. Similarly the cancer button is for medical applications pertaining towards cancer detection and analysis. This is for later development as well.

Upon selecting the Corrosion button we more on to the Structural health assessment page in the software.

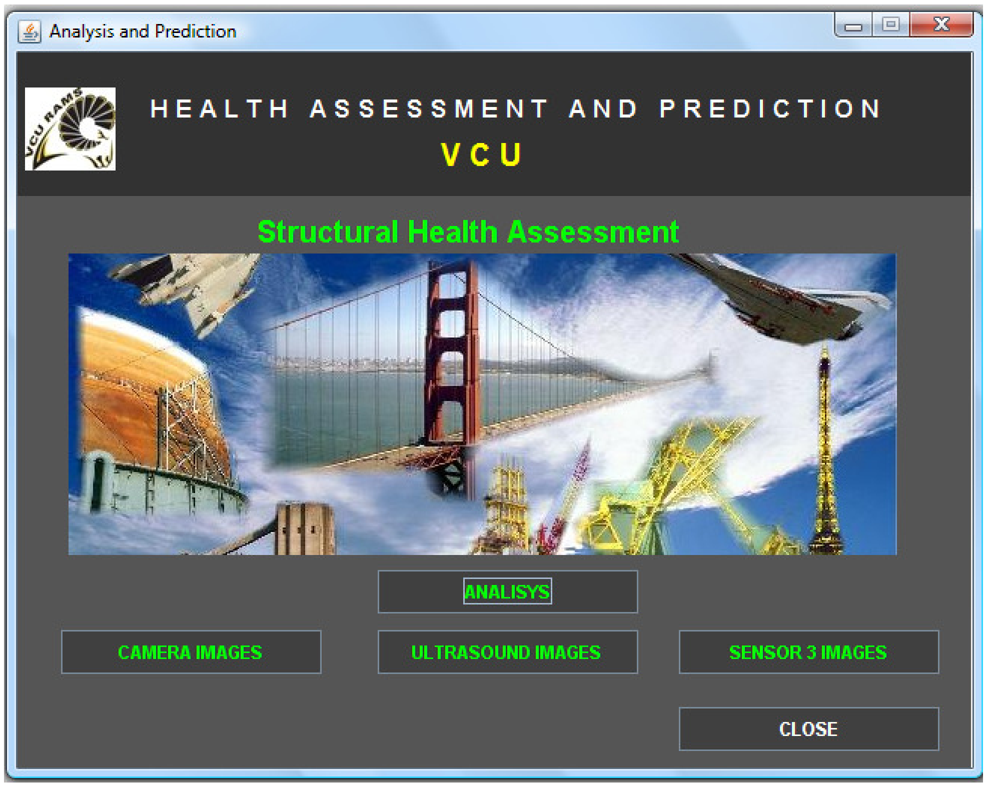


Figure 16: Corrosion Assessment Page

Here we can choose the various sensor devices and its captured images and send it through analysis. As per the design setup the images which were captured from various sensors from remote locations are relayed and stored in the database on the server system. From here we get to choose the images by clicking on the type of image we want and then sending them for processing.

Suppose the camera images button is chosen then we move to the browse page where the subsequent directories are allowed to be browsed for selection and analysis.

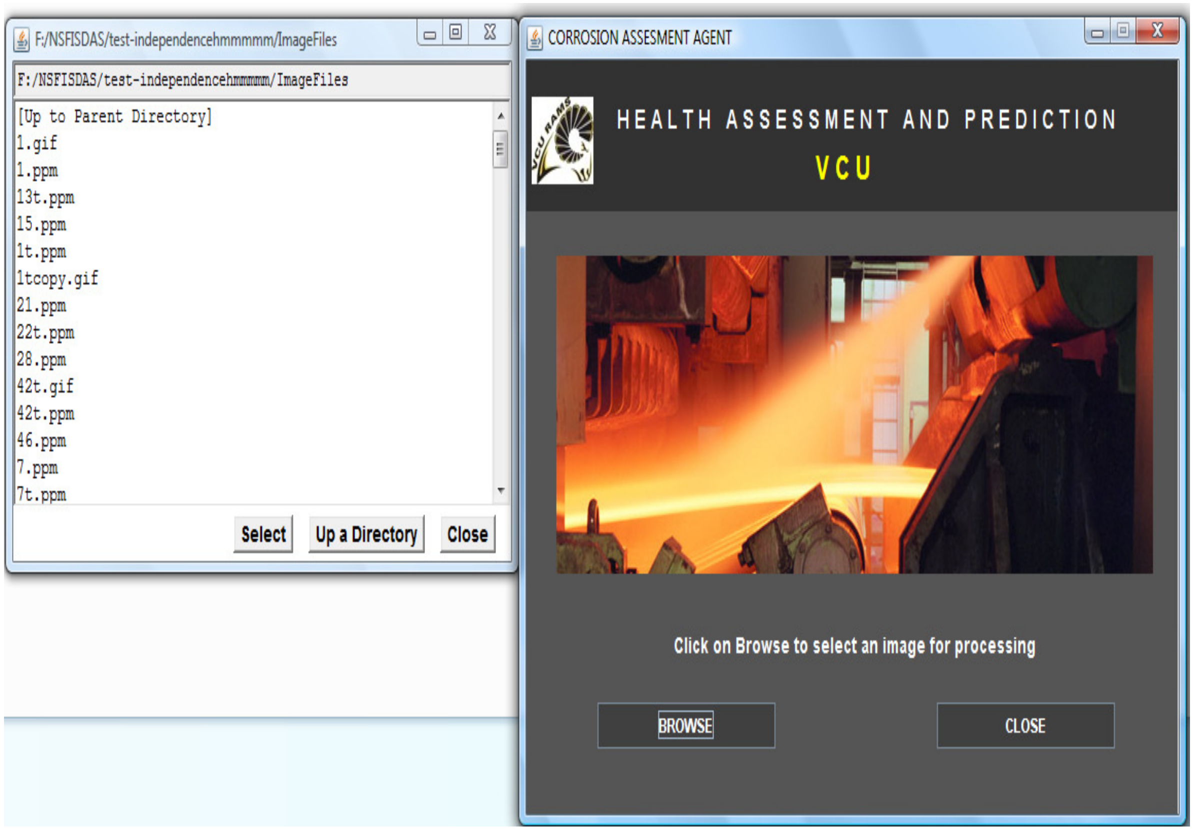

F:/NSFISDAS/test-independencehmmmmm/ImageFiles
F:/NSFISDAS/test-independencehmmmmm/ImageFiles
[Up to Parent Directory]
1.gif
1.ppm
13t.ppm
15.ppm
1t.ppm
1tcopy.gif
21.ppm
22t.ppm
28.ppm
42t.gif
42t.ppm
46.ppm
7.ppm
7t.ppm
Select
Up a Directory
Close
CORROSION ASSESMENT AGENT
HEALTH ASSESSMENT AND PREDICTION
VCU
Click on Browse to select an image for processing
BROWSE
CLOSE

Figure 17: Browse Page

Once we have selected the image to be analyzed, it is sent through a set of procedures for damage identification and quantification. The overall process of identification and quantification is essentially involves two stages, first being the classification of various regions in the image as corroded or un-corroded. Second being the prediction of the material loss of the corroded regions.

The classification process involves segmenting the image into various regions. Multi-resolution wavelet analysis is performed on the NDI images to obtain a set of wavelet coefficients as feature vectors. These features will be used for the identification of the damaged regions on the images using clustering techniques.

Each of the segments on the segmented image would correspond to a damaged region or an undamaged region. There are several algorithms available for segmentation process based on images. Amongst these however texture based segmentation on wavelets is considered a powerful technique. After the image is segmented, histogram features are extracted from each segment.

After the damaged segments are indentified, a number of features are extracted from each identified segment and a back propagation neural network is used to quantify the damage. Neural networks are capable of realizing a variety of nonlinear relationships of considerable complexity and are effectively used in this research. The quantification of damage is based on the extent material loss.

After sending it through the image based analysis steps and procedures the software then shows the images that were produced during processed such as original image, segmented image, identified regions image and wavelet transforms image.

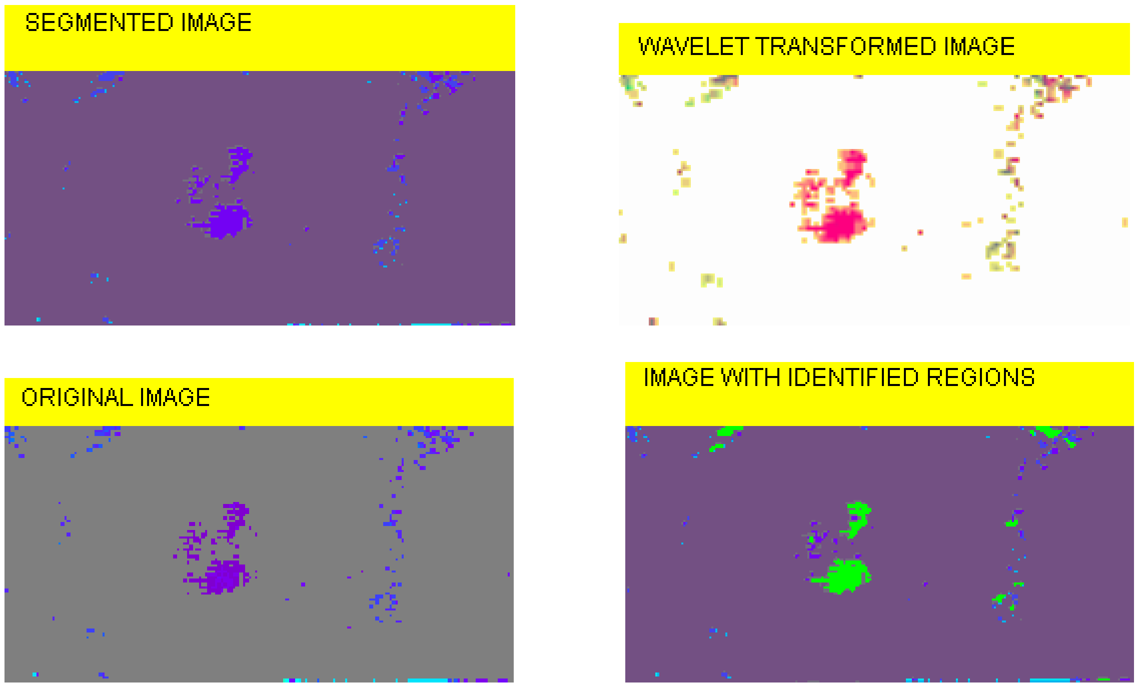


Figure 18: Processing stages of Input Image

These images along with some quantified values can be used to identify and quantify the corrosion of the metal surfaces that are being analyzed. The personnel analyzing these images can then make a decision on what is has to be done with the metal surface under analysis.

5.1 Installing the Java Virtual Machine on the gumstix

A programming environment had to be setup on the gumstix so as to be able to write target application programs that could be run on the gumstix as well. For this java was the proffered the programming language, due to its graphical interface development and its ease of programming.

To setup Java on the gumstix a proper java environment was required to be able to run java virtual machine on the gumstix architecture. For the Gumstix a recommended java virtual machine is "JamVM". It is small and very efficient for the arm architecture.

These were the following ipkg packages that were installed onto the gumstix kernel to setup a java virtual machine and programming environment onto it;

```
  'classpath_0.96.1-r2_armv5te.ipk  '
'classpath-common_0.96.1-r2_armv5te.ipk'
  jamvm-dev_1.5.0-r0_armv5te.ipk
jamvm-dbg_1.5.0-r0_armv5te.ipk
jikes_1.22-r1_armv5te.ipk
classpath-dbg_0.96.1-r2_armv5te.ipk
classpath-dev_0.96.1-r2_armv5te.ipk
classpath-doc_0.96.1-r2_armv5te.ipk
classpath-examples_0.96.1-r2_armv5te.ipk
classpath-gconf_0.96.1-r2_armv5te.ipk
classpath-gtk_0.96.1-r2_armv5te.ipk
classpath-tools-doc_0.96.1-r2_armv5te.ipk
classpath-tools_0.96.1-r2_armv5te.ipk
```

This enabled us to get java programs running on gumstix handheld unit.

6. Server Side Configuration and Data Upload Access

The server side configuration primarily involves the method in which the data that is being relayed by the handheld device is stored on the server machine for processing them with the computational analysis software. The idea is to facilitate real time upload of data which is initiated from the handheld device as and when it captures a new image or related data from its sensors.

To enable such kind of real time uploads from the handheld device a web-server was setup on the server machine. A web server is a computer program that is responsible for accepting HTTP requests from web clients, which are known as web browsers, and serving them HTTP responses along with optional data contents, which usually are web pages such as HTML documents and linked objects (images, etc.).

The scripting for such a server setup is done with PHP. PHP is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It generally runs on a web server, taking PHP code as its input and creating web pages as output. It can be deployed on most web servers and on almost every operating system and platform free of charge.

The server machine that has been setup for experimental purpose for this system is a Unix based Ubuntu Linux machine. The Ubuntu OS has a webserver in it by default. The default webserver in it was Apache.

7. Future Work and Conclusion

There is a broad range of applications that these kinds of a wireless handheld sensor system can be implemented in. For experimental purposes it was chose to first run trial analysis with corrosion detection on metal surfaces. The test are planned to be conducted on certain bridges in Virginia. Based on these test and its corresponding success, the next plan is to extend the application of this device onto other fields.

One big area of interest is in the field of medical application. The idea is to enable the handheld unit with x sensors capable of capturing images of bone and cancer. Thus considering a scenario of a accident victim the developed handheld device can be used at the site of the accident itself by the ambulance personal, and then relay the images captured to the nearest hospital where the doctor get to study and analyze the patients condition without the patient having to be there. Thus enabling the doctor to get right into helping the patient out when he does arrive.

Similar plans of application can be put forward in cancer detection and monitoring. Ultimately the system can be implemented where ever remote analysis and data retrieval is required. Thus the intention of this project is to build a strong foundation for developing different kinds of such handheld unit capable of performing several tasks.

Literature Cited

Literature Cited

[1] Bartelds, G. (1998). Aircraft structural health monitoring, prospects for smart solutions from a European
viewpoint. Journal of Intelligent Material Systems and Structures, 9, 906–910.

[2] Finlayson, R. D.; Friesel, M. A.; Carlos, M. F.; Miller, R.; Godinez, V. (2000): Acoustic Emission Structural Health Management Systems (AE-SHMS). Proceedings of SPIE - The International Society for Optical Engineering, v 3994, pp. 128-137.

[3] Haugse, E.; Leeks, T.; Ikegami, R.; Johnson, P.; Ziola, S.; Dorighi, J.; May, S.; Phelps, N. (1999): Crack growth detection and monitoring using broadband acoustic emission techniques. Proceedings of SPIE - The International Society for Optical Engineering, vol. 3586, pp. 32-40.

[4] Rong-Sheng, G. (2004): Evaluation of calendar damage of aircraft structures using acoustic emission. Key Engineering Materials, v 270-273, No. I, pp. 503-509.

[5] Wilson, D. S.; Hagemaier, D. J. (1999): ABCs of NDT development and adoption for aging aircraft. Materials Evaluation, vol. 57, No. 3, pp. 336-346.

[6] Reda Taha, M. M., Noureldin, A., Lucero, J. L. and Baca, T. J. (2006) Wavelet Transform for Structural Health Monitoring: A Compendium of Uses and Features. Structural Health Monitoring, 5: 267-295.

[7] J.Wang, M. Mbonisi and J. Liu, ``Supporting Video Data in Wireless Sensor Networks," IEEE International Symposium on Multimedia, 2007.

[8] J.Wang and J. Liu, ``Uplink Relaying in Wireless Networks with Out-of-cell Interference Reduction," ACM Journal of Wireless Network, accepted.

[9] ZigBee™/IEEE 802.15.4-Transceiver AT86RF230, Datasheet

[10] Chopard, B. and Droz, M. (1998). Cellular Automata
Modeling of Physical Systems, Cambridge University
Press, England.

[11] Malik, J. and Perona, P. (1990). Preattentive texture
discrimination with early vision mechanisms. J. Opt.
Soc. Am. Series, A7, 923–932.

[12] Turner, M.R. (1986). Texture discrimination by Gabor

functions. Biol. Cybern., 55, 71–82.
[13] M.J. Palak, R. M. V. Pidaparti, S. Rebbapragada, Intelligent computational methods for corrosion damage assessment, AIAA journal, Vol. 39, No. 10, October 2001.

[14] C328-7640 Data Sheet

[15] ATmega1280/V, ATmega1281/V datasheet

[16] Sierra Wireless AIRCARD® 595U USB MODEM

APPENDIX A

Code for homepage.java

```
import java.awt.Toolkit;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
//import java.applet.*;
import com.borland.jbcl.layout.*;
import java.io.*;
import java.net.*;
import java.awt.image.*;
import javax.imageio.*;


public class homePage extends JFrame
{//MainIsdas

     boolean isStandalone = false;
     int checkFlag = 2;

     //Required For Header.
    JPanel headerPanel = new JPanel();
    JLabel isdasLabel = new JLabel();
    JLabel corrLabel = new JLabel();
    JLabel VCULabel = new JLabel();
    JButton imageButton = new JButton();
    JButton analyticalButton = new JButton();
    JPanel cardPanel = new JPanel();
    CardLayout cardLayout1 = new CardLayout();

    //Required for Image Panel.

    Button originalImageButton = new Button();
    JPanel imagePanel = new JPanel();
    JPanel analyticalPanel = new JPanel();
    JLabel titleLabel = new JLabel();
    JTextField matlTypeText = new JTextField();
    JLabel deviceLabel = new JLabel();
```

```
JComboBox deviceChoice = new JComboBox();
JLabel frequencyLabel = new JLabel();
JTextField frequencyText = new JTextField();
JLabel filenameLabel = new JLabel();
public JTextField filenameText = new JTextField();
JButton browseButton = new JButton();
JLabel imageTypeLabel = new JLabel();
JLabel textLabel = new JLabel();
JButton processButton = new JButton();
JButton exitButton = new JButton();
JButton corrosionButton = new JButton();
JButton boneButton = new JButton();
JButton cancerButton = new JButton();
JCheckBox panelCheckBox = new JCheckBox();
CheckboxGroup imageCheckboxGroup = new CheckboxGroup();
JCheckBox corrodeCheckBox = new JCheckBox();
JButton cameraButton = new JButton();

ImageIcon icon = new ImageIcon("newimg.jpg",
                "a pretty but meaningless splat");

JLabel imageLabel = new JLabel("", icon, JLabel.CENTER);
ImageIcon logo = new ImageIcon("logo.jpg",
                        "VCUlogo");

JLabel logoLabel = new JLabel("", logo,JLabel.LEFT);

XYLayout xYLayout1 = new XYLayout();
XYLayout xYLayout2 = new XYLayout();
XYLayout xYLayout3 = new XYLayout();
XYLayout xYLayout4 = new XYLayout();

//stuff for testFrame computation
private String fileName;
 private BufferedReader input;
 private PpmReader ppmhandler;
 private int height, width;
 private ImageMatrix OriginalImage;
 private ImageMatrix SegmentedImage;
 private BufferedImage image;
 private MarkSegments markSegments;
 private String catString;
```

```
             private float waveMat[][];
            private float MyImage[][];
            private ImageMatrix LLImage;


//Begin of constructor Check
            public homePage(String file, String catString)
            {
                        super(" Health Assessment Agent");

            }//End of contructor check


            //MAIN Method
            public static void main(String m[])
            {
                         new homePage();
            }


            //Construct the application
            public homePage() {
                         super("Analysis and Prediction");


                         setResizable(false);
                         init();
                         setVisible(true);
                         setSize(640,500);//pack();
            }

            //Initialize the application
            public void init() {
              try  {
                jbInit();
              }
              catch(Exception e)  {
                e.printStackTrace();
              }
            }

                        //Component initialization
```

```
        private void jbInit() throws Exception {

                Color headColor = new Color(50,50,50);

            //Header Initialization.

            corrLabel.setBackground(Color.gray);
            corrLabel.setForeground(Color.white);
            corrLabel.setFont(new java.awt.Font("Dialog", 1, 17));
            corrLabel.setName("corrLabel");
            corrLabel.setAlignmentX(1);
            corrLabel.setText("H E A L T H   A S S E S S M E N T   A N D   P R E D I C T I
O N");
            corrLabel.setVerticalAlignment(SwingConstants.CENTER);
            this.setBackground(Color.GRAY);
            this.setLayout(xYLayout4);
            headerPanel.setBackground(headColor);
            headerPanel.setName("headerPanel");
            headerPanel.setLayout(xYLayout3);

            VCULabel.setBackground(Color.gray);
            VCULabel.setForeground(Color.yellow);
            VCULabel.setFont(new java.awt.Font("Dialog", 1, 20));
            VCULabel.setName("corrLabel");
            VCULabel.setAlignmentX(1);
            VCULabel.setText("                    V C U");
            VCULabel.setVerticalAlignment(SwingConstants.CENTER);


                titleLabel.setBackground(Color.gray);
                titleLabel.setForeground(Color.green);
                titleLabel.setFont(new java.awt.Font("Dialog", 1, 20));
                titleLabel.setName("titleLabel");
                titleLabel.setAlignmentX(1);
                titleLabel.setText("Wireless Sensor And Computational Intelligence");
             titleLabel.setVerticalAlignment(SwingConstants.CENTER);

    Color bodyColor = new Color(85,85,85);
                 // Card Panel Initialization.

    cardPanel.setBackground(Color.lightGray);
    cardPanel.setName("cardPanel");
    cardPanel.setLayout(cardLayout1);

    //Card Panel comprises of two panels : Image Panel and Analytical Panel.

    imagePanel.setBackground(bodyColor);
    imagePanel.setName("imagePanel");
    imagePanel.setLayout(xYLayout1);


        textLabel.setOpaque(true);
    textLabel.setBackground(null);
    textLabel.setFont(new Font("Dialog", 1, 14));
    textLabel.setForeground(Color.white);
    textLabel.setName("textLabel");
    //textLabel.setAlignmentX(1);
    textLabel.setText(" Click on Browse to select an image for processing");

    browseButton.setBackground(Color.darkGray);
    browseButton.setFont(new Font("Dialog", 1, 12));
    browseButton.setForeground(Color.white);
    browseButton.setName("browseButton");
    browseButton.setLabel("BROWSE");
    browseButton.addActionListener(new homePage_browseButton_actionAdapter(this));

        exitButton.setBackground(Color.darkGray);
    exitButton.setFont(new Font("Dialog", 1, 12));
    exitButton.setForeground(Color.white);
    exitButton.setName("exitButton");
    exitButton.setLabel("EXIT");
    exitButton.addActionListener(new homePage_exitButton_actionAdapter(this));

    corrosionButton.setBackground(Color.darkGray);
        corrosionButton.setFont(new Font("Dialog", 1, 12));
        corrosionButton.setForeground(Color.white);
        corrosionButton.setName("corrosionButton");
        corrosionButton.setLabel("CORROSION");
   corrosionButton.addActionListener(new
homePage_corrosionButton_actionAdapter(this));

    boneButton.setBackground(Color.darkGray);
            boneButton.setFont(new Font("Dialog", 1, 12));
            boneButton.setForeground(Color.white);
```

```
                boneButton.setName("boneButton");
                boneButton.setLabel("BONE");
   boneButton.addActionListener(new homePage_boneButton_actionAdapter(this));

   cancerButton.setBackground(Color.darkGray);
                        cancerButton.setFont(new Font("Dialog", 1, 12));
                        cancerButton.setForeground(Color.white);
                        cancerButton.setName("cancerButton");
                        cancerButton.setLabel("CANCER");
   cancerButton.addActionListener(new homePage_cancerButton_actionAdapter(this));


         originalImageButton.setBackground(Color.black);
   originalImageButton.setFont(new java.awt.Font("Dialog", 1, 12));
   originalImageButton.setForeground(Color.green);
   originalImageButton.setName("originalImageButton");
   originalImageButton.setLabel("Original Image");
  //originalImageButton.addActionListener(new
homePage_originalImageButton_actionAdapter(this));

                originalImageButton.addActionListener(new
java.awt.event.ActionListener()
                   {
                     public void actionPerformed(ActionEvent e)
                     {
                      //originalImageButton_actionPerformed(e);
                      DisplayImage display = new
DisplayImage("Original_"+fileName,OriginalImage);


                     }
                });

   xYLayout4.setHeight(510); //BELLE
   xYLayout4.setWidth(628);

   this.add(headerPanel, new XYConstraints(-1, 0, 634, 95));
   //headerPanel.add(isdasLabel, new XYConstraints(89, 1, 453, 23));
   headerPanel.add(corrLabel, new XYConstraints(89, 22, 500, 28));
   headerPanel.add(VCULabel, new XYConstraints(150, 55, 453, 24));
    headerPanel.add(logoLabel, new XYConstraints(6, 20, 60, 60));
   //headerPanel.add(analyticalButton, new XYConstraints(347, 57, 127, 29));
   //headerPanel.add(imageButton, new XYConstraints(140, 58, 119, 29));

   this.add(cardPanel, new XYConstraints(-1, 83, 634, 425));

   cardPanel.add(imagePanel, "imagePanel");
         //imagePanel.add(browseButton, new XYConstraints(70, 321, 173, 29));
   imagePanel.add(exitButton, new XYConstraints(400, 321, 173, 29));
   imagePanel.add(corrosionButton, new XYConstraints(310, 120, 173, 29));
   imagePanel.add(boneButton, new XYConstraints(310, 170, 173, 29));
   imagePanel.add(cancerButton, new XYConstraints(310, 230, 173, 29));
   //imagePanel.add(originalImageButton, new XYConstraints(91, 88, 196, 31));
   imagePanel.add(titleLabel, new XYConstraints(91, 20, 600, 31));
   //imagePanel.add(textLabel, new XYConstraints(140, 270, 360, 29));
   imagePanel.add(imageLabel, new XYConstraints(30,60,250,300));


  }// End of Class check


  void browseButton_actionPerformed(ActionEvent e)
  {
                //System.out.println("Stop clicking me");
                String[] args = new String[]{};
                RunFileLister rfl = new RunFileLister(args);
                Thread t = new Thread(rfl);
                t.start();
   }

void originalImageButton_actionPerformed(ActionEvent e)
  {
                DisplayImage display;
                display = new DisplayImage("Original_"+fileName,OriginalImage);
                System.out.println("HELLO Original");
   }

  void exitButton_actionPerformed(ActionEvent e)
  {
         //this.dispose();
                System.exit(0);
  }

         void corrosionButton_actionPerformed(ActionEvent e)
```

```
   {
          corrosionHome CH = new corrosionHome();
   }

  void boneButton_actionPerformed(ActionEvent e)
   {
          bone b = new bone();
   }

   void cancerButton_actionPerformed(ActionEvent e)
         {
                 cancer c = new cancer();
   }

}//end of MainISdas

// Image Panel Related Classes.

class homePage_originalImageButton_actionAdapter implements
java.awt.event.ActionListener
 {
                homePage adaptee;
                homePage_originalImageButton_actionAdapter(homePage adaptee)
                {
                   this.adaptee = adaptee;
                }
     public void actionPerformed(ActionEvent e)
           {
              adaptee.originalImageButton_actionPerformed(e);
           }
 }

class homePage_browseButton_actionAdapter implements java.awt.event.ActionListener
{
  homePage adaptee;

  homePage_browseButton_actionAdapter(homePage adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.browseButton_actionPerformed(e);
  }
}

class homePage_corrosionButton_actionAdapter implements
java.awt.event.ActionListener
{
  homePage adaptee;

  homePage_corrosionButton_actionAdapter(homePage adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.corrosionButton_actionPerformed(e);
  }
}

class homePage_boneButton_actionAdapter implements java.awt.event.ActionListener
{
  homePage adaptee;

  homePage_boneButton_actionAdapter(homePage adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.boneButton_actionPerformed(e);
  }
}

class homePage_cancerButton_actionAdapter implements java.awt.event.ActionListener
{
  homePage adaptee;

  homePage_cancerButton_actionAdapter(homePage adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.cancerButton_actionPerformed(e);
  }
}


class homePage_exitButton_actionAdapter implements java.awt.event.ActionListener
{
  homePage adaptee;

  homePage_exitButton_actionAdapter(homePage adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.exitButton_actionPerformed(e);
  }
}
```

Code for Corrosionhome.java

```
import java.awt.Toolkit;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
//import java.applet.*;
import com.borland.jbcl.layout.*;
import java.io.*;
import java.net.*;
import java.awt.image.*;
import javax.imageio.*;


public class corrosionHome extends JFrame
{//MainIsdas

         boolean isStandalone = false;
         int checkFlag = 2;

         //Required For Header.
        JPanel headerPanel = new JPanel();
        JLabel isdasLabel = new JLabel();
        JLabel corrLabel = new JLabel();
        JLabel VCULabel = new JLabel();
        JButton imageButton = new JButton();
        JButton analyticalButton = new JButton();
        JPanel cardPanel = new JPanel();
        CardLayout cardLayout1 = new CardLayout();

        //Required for Image Panel.

        Button originalImageButton = new Button();
        JPanel imagePanel = new JPanel();
        JPanel analyticalPanel = new JPanel();
        JLabel titleLabel = new JLabel();
        JTextField matlTypeText = new JTextField();
        JLabel deviceLabel = new JLabel();
        JComboBox deviceChoice = new JComboBox();
        JLabel frequencyLabel = new JLabel();
        JTextField frequencyText = new JTextField();
        JLabel filenameLabel = new JLabel();
```

```
public JTextField filenameText = new JTextField();
JButton browseButton = new JButton();
JLabel imageTypeLabel = new JLabel();
JLabel textLabel = new JLabel();
JButton processButton = new JButton();
JButton exitButton = new JButton();
JButton analyseButton = new JButton();
JButton cameraButton = new JButton();
JButton boneButton = new JButton();
JButton cancerButton = new JButton();
JCheckBox panelCheckBox = new JCheckBox();
CheckboxGroup imageCheckboxGroup = new CheckboxGroup();
JCheckBox corrodeCheckBox = new JCheckBox();
//JButton cameraButton = new JButton();

ImageIcon icon = new ImageIcon("Structure.jpg",
                 "a pretty but meaningless splat");

JLabel imageLabel = new JLabel("", icon, JLabel.CENTER);
ImageIcon logo = new ImageIcon("logo.jpg",
                          "VCUlogo");

JLabel logoLabel = new JLabel("", logo,JLabel.LEFT);

XYLayout xYLayout1 = new XYLayout();
XYLayout xYLayout2 = new XYLayout();
XYLayout xYLayout3 = new XYLayout();
XYLayout xYLayout4 = new XYLayout();

//stuff for testFrame computation
private String fileName;
 private BufferedReader input;
 private PpmReader ppmhandler;
 private int height, width;
 private ImageMatrix OriginalImage;
 private ImageMatrix SegmentedImage;
 private BufferedImage image;
 private MarkSegments markSegments;
 private String catString;
 private float waveMat[][];
private float MyImage[][];
private ImageMatrix LLImage;
```

```
//Begin of constructor Check
        public corrosionHome(String file, String catString)
        {
                super(" Health Assessment Agent");

                new corrosionHome();
                // setResizable(false);
                //                  init();
                //                  setVisible(true);
                 // setSize(640,500);//pack();




                 //belle

        }//End of contructor check


        //MAIN Method
        public static void main(String m[])
        {
                 new corrosionHome();
        }


        //Construct the application
        public corrosionHome() {
                 super("Analysis and Prediction");


                 setResizable(false);
                 init();
                 setVisible(true);
                 setSize(640,500);//pack();
        }

        //Initialize the application
        public void init() {
          try  {
```

```
    jbInit();
  }
  catch(Exception e)  {
    e.printStackTrace();
  }
}

        //Component initialization
private void jbInit() throws Exception {

        Color headColor = new Color(50,50,50);

  //Header Initialization.

  corrLabel.setBackground(Color.gray);
  corrLabel.setForeground(Color.white);
  corrLabel.setFont(new java.awt.Font("Dialog", 1, 17));
  corrLabel.setName("corrLabel");
  corrLabel.setAlignmentX(1);
  corrLabel.setText("H E A L T H   A S S E S S M E N T   A N D   P R E D I C T I
O N");
  corrLabel.setVerticalAlignment(SwingConstants.CENTER);
  this.setBackground(Color.GRAY);
  this.setLayout(xYLayout4);
  headerPanel.setBackground(headColor);
  headerPanel.setName("headerPanel");
  headerPanel.setLayout(xYLayout3);

  VCULabel.setBackground(Color.gray);
  VCULabel.setForeground(Color.yellow);
  VCULabel.setFont(new java.awt.Font("Dialog", 1, 20));
  VCULabel.setName("corrLabel");
  VCULabel.setAlignmentX(1);
  VCULabel.setText("                    V C U");
  VCULabel.setVerticalAlignment(SwingConstants.CENTER);


        titleLabel.setBackground(Color.gray);
        titleLabel.setForeground(Color.green);
        titleLabel.setFont(new java.awt.Font("Dialog", 1, 20));
        titleLabel.setName("titleLabel");
        titleLabel.setAlignmentX(1);
```

```
                    titleLabel.setText("Structural Health Assessment");
                titleLabel.setVerticalAlignment(SwingConstants.CENTER);

    Color bodyColor = new Color(85,85,85);
                    // Card Panel Initialization.

    cardPanel.setBackground(Color.lightGray);
    cardPanel.setName("cardPanel");
    cardPanel.setLayout(cardLayout1);

    //Card Panel comprises of two panels : Image Panel and Analytical Panel.

    imagePanel.setBackground(bodyColor);
    imagePanel.setName("imagePanel");
    imagePanel.setLayout(xYLayout1);


        textLabel.setOpaque(true);
    textLabel.setBackground(null);
    textLabel.setFont(new Font("Dialog", 1, 14));
    textLabel.setForeground(Color.white);
    textLabel.setName("textLabel");
    //textLabel.setAlignmentX(1);
    textLabel.setText(" Click on Browse to select an image for processing");

    browseButton.setBackground(Color.darkGray);
    browseButton.setFont(new Font("Dialog", 1, 12));
    browseButton.setForeground(Color.white);
    browseButton.setName("browseButton");
    browseButton.setLabel("BROWSE");
    browseButton.addActionListener(new
corrosionHome_browseButton_actionAdapter(this));

        exitButton.setBackground(Color.darkGray);
    exitButton.setFont(new Font("Dialog", 1, 12));
    exitButton.setForeground(Color.white);
    exitButton.setName("exitButton");
    exitButton.setLabel("CLOSE");
    exitButton.addActionListener(new corrosionHome_exitButton_actionAdapter(this));

    cameraButton.setBackground(Color.darkGray);
        cameraButton.setFont(new Font("Dialog", 1, 12));
        cameraButton.setForeground(Color.green);
        cameraButton.setName("sensor1Button");
        cameraButton.setLabel("CAMERA IMAGES");
    cameraButton.addActionListener(new
corrosionHome_cameraButton_actionAdapter(this));

    analyseButton.setBackground(Color.darkGray);
                    analyseButton.setFont(new Font("Dialog", 1, 12));
                    analyseButton.setForeground(Color.green);
                    analyseButton.setName("analyseButton");
                    analyseButton.setLabel("ANALISYS");
     analyseButton.addActionListener(new
corrosionHome_analyseButton_actionAdapter(this));

    boneButton.setBackground(Color.darkGray);
                    boneButton.setFont(new Font("Dialog", 1, 12));
                    boneButton.setForeground(Color.green);
                    boneButton.setName("sensor2Button");
                    boneButton.setLabel("ULTRASOUND IMAGES");
    //boneButton.addActionListener(new corrosionHome_exitButton_actionAdapter(this));

    cancerButton.setBackground(Color.darkGray);
                                cancerButton.setFont(new Font("Dialog", 1, 12));
                                cancerButton.setForeground(Color.green);
                                cancerButton.setName("sensor3Button");
                                cancerButton.setLabel("SENSOR 3 IMAGES");
    //cancerButton.addActionListener(new
corrosionHome_exitButton_actionAdapter(this));


        originalImageButton.setBackground(Color.black);
    originalImageButton.setFont(new java.awt.Font("Dialog", 1, 12));
    originalImageButton.setForeground(Color.green);
    originalImageButton.setName("originalImageButton");
    originalImageButton.setLabel("Original Image");
   //originalImageButton.addActionListener(new
corrosionHome_originalImageButton_actionAdapter(this));

                    originalImageButton.addActionListener(new
java.awt.event.ActionListener()
                        {
                          public void actionPerformed(ActionEvent e)
                          {
                           //originalImageButton_actionPerformed(e);
                          DisplayImage display = new
DisplayImage("Original_"+fileName,OriginalImage);


                          }
                      });

    xYLayout4.setHeight(510); //BELLE
    xYLayout4.setWidth(628);

    this.add(headerPanel, new XYConstraints(-1, 0, 634, 95));
    //headerPanel.add(isdasLabel, new XYConstraints(89, 1, 453, 23));
    headerPanel.add(corrLabel, new XYConstraints(89, 22, 500, 28));
    headerPanel.add(VCULabel, new XYConstraints(150, 55, 453, 24));
     headerPanel.add(logoLabel, new XYConstraints(6, 20, 60, 60));
    //headerPanel.add(analyticalButton, new XYConstraints(347, 57, 127, 29));
    //headerPanel.add(imageButton, new XYConstraints(140, 58, 119, 29));

    this.add(cardPanel, new XYConstraints(-1, 83, 634, 425));

    cardPanel.add(imagePanel, "imagePanel");
        //imagePanel.add(browseButton, new XYConstraints(70, 321, 173, 29));
    imagePanel.add(exitButton, new XYConstraints(440, 351, 173, 29));
    imagePanel.add(cameraButton, new XYConstraints(30, 300, 173, 29));
    imagePanel.add(analyseButton, new XYConstraints(240, 260, 173, 29));
    imagePanel.add(boneButton, new XYConstraints(240, 300, 173, 29));
    imagePanel.add(cancerButton, new XYConstraints(440, 300, 173, 29));
    //imagePanel.add(originalImageButton, new XYConstraints(91, 88, 196, 31));
    imagePanel.add(titleLabel, new XYConstraints(160, 20, 600, 31));
    //imagePanel.add(textLabel, new XYConstraints(140, 270, 360, 29));
    //imagePanel.add(imageLabel, new XYConstraints(150,60,300,230));
    imagePanel.add(imageLabel, new XYConstraints(10,50,600,200));


  }// End of Class check


  void browseButton_actionPerformed(ActionEvent e)
  {
                    //System.out.println("Stop clicking me");
```

```
                    String[] args = new String[]{};
                    RunFileLister rfl = new RunFileLister(args);
                    Thread t = new Thread(rfl);
                    t.start();
   }

void originalImageButton_actionPerformed(ActionEvent e)
  {
                    DisplayImage display;
                    display = new DisplayImage("Original_"+fileName,OriginalImage);
                    System.out.println("HELLO Original");
   }

          void cameraButton_actionPerformed(ActionEvent e)
    {
            MainIsdas M = new MainIsdas();
    }

    void analyseButton_actionPerformed(ActionEvent e)
             {
                      analyse A = new analyse();
    }

  void exitButton_actionPerformed(ActionEvent e)
  {
            this.dispose();
                    //System.exit(0);
  }

}//end of MainISdas

// Image Panel Related Classes.

class corrosionHome_originalImageButton_actionAdapter implements
java.awt.event.ActionListener
 {
                    corrosionHome adaptee;
                    corrosionHome_originalImageButton_actionAdapter(corrosionHome
adaptee)
                    {
                       this.adaptee = adaptee;
                    }
```

```
      public void actionPerformed(ActionEvent e)
              {
                 adaptee.originalImageButton_actionPerformed(e);
              }
 }

class corrosionHome_browseButton_actionAdapter implements
java.awt.event.ActionListener
{
  corrosionHome adaptee;

  corrosionHome_browseButton_actionAdapter(corrosionHome adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.browseButton_actionPerformed(e);
  }
}

class corrosionHome_analyseButton_actionAdapter implements
java.awt.event.ActionListener
{
  corrosionHome adaptee;

  corrosionHome_analyseButton_actionAdapter(corrosionHome adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.analyseButton_actionPerformed(e);
  }
}


class corrosionHome_cameraButton_actionAdapter implements
java.awt.event.ActionListener
{
  corrosionHome adaptee;

  corrosionHome_cameraButton_actionAdapter(corrosionHome adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.cameraButton_actionPerformed(e);
  }
}
class corrosionHome_exitButton_actionAdapter implements java.awt.event.ActionListener
{
  corrosionHome adaptee;

  corrosionHome_exitButton_actionAdapter(corrosionHome adaptee)
  {
    this.adaptee = adaptee;
  }

  public void actionPerformed(ActionEvent e)
  {
    adaptee.exitButton_actionPerformed(e);
  }
}
```

Code for Capture.sh

```
#!/bin/bash
#
#
# ------------------------------------------------------------------------
# # Shell scrip to send upload a file online using curl command.
# ------------------------------------------------------------------------
uvccapture
curl -F ufile=@snap.jpg -F Submit=Upload http://128.172.192.82:3310/upload_ac.php
 echo "done"
```

Code of the Java Program on the Gumstix

```
import java.awt.BorderLayout;
import java.awt.Container;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.InputEvent;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.SwingUtilities;
import java.io.*;

public class ButtonActionSample {

  public static void main(String args[]) throws Exception
{




    JFrame frame = new JFrame("Button Sample");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JButton button = new JButton("Select Me");




    ActionListener actionListener = new ActionListener() {
      public void actionPerformed(ActionEvent actionEvent) {
        System.out.println("I was selected.");
/* try {
Process p = Runtime.getRuntime().exec("cmd /C dir"); } catch (IOException e) {
            e.printStackTrace();}
                        System.out.println("done");

  }*/
      try {

                        Process p = Runtime.getRuntime().exec("capture.sh");
            //       Process p1 = Runtime.getRuntime().exec("curl -F ufile=@snap.jpg -F
Submit=Upload http://%s/upload_ac.php");
                     System.out.println("Snap has been taken and uploaded online! ");
//PrintWriter pw = new PrintWriter(new OutputStreamWriter(p1.getOutputStream()));

            } catch (IOException e) {
                     e.printStackTrace();
}
}
   };
   button.addActionListener(actionListener);


   Container contentPane = frame.getContentPane();
   contentPane.add(button, BorderLayout.CENTER);
   frame.setSize(300, 100);
   frame.setVisible(true);
  }
}
```