

Improving Greenhouse's Automation and Data Acquisition with Mobile Robot Controlled system via Wireless Sensor Network

István Matijevics* and Simon János**

**University of Szeged, Institute of Informatics
Hungary*

***Subotica Tech, Department of Informatics
Serbia*

1. Introduction

The function of a greenhouse is to create the optimal growing conditions for the full lifecycle of the plants. Using autonomous measuring stations helps to monitor all the necessary parameters for creating the optimal environment in the greenhouse. The robot equipped with sensors is capable of driving to the end and back along crop rows inside the greenhouse. This chapter deals with the implementation of mobile measuring station in greenhouse environment. It introduces a wireless sensor network that was used for the purpose of measuring and controlling the greenhouse application. Continuous advancements in wireless technology and miniaturization have made the deployment of sensor networks to monitor various aspects of the environment increasingly flexible. Climate monitoring is vitally important to the operation in greenhouses and the quality of the collected information has a great influence on the precision and accuracy of control results. Currently, the agro-alimentary market field incorporates diverse data acquisition techniques. Normally, the type of acquisition system is chosen to be optimal for the control algorithm to be used. For traditional climate monitoring and control systems, all sensors are distributed through the greenhouse and connected to the device performing the control tasks. These equipments use time-based data sampling techniques as a consequence of using time-based controllers. Typical applications of WSNs include monitoring, tracking, and controlling. Some of the specific applications are habitat monitoring, object tracking, etc. In a typical application, a WSN is scattered in a region where it is meant to collect data through its sensor node. The WSN-based controller has allowed a considerable decrease in the number of changes in the control action and made possible a study of the compromise between quantity of transmission and control performance. In modern greenhouses, several measurement points are required to trace down the local climate parameters in different parts of the big greenhouse to make the greenhouse automation system work properly. Cabling would make the measurement system expensive and vulnerable. Moreover, the cabled measurement points are difficult to relocate once they are installed. Thus, a wireless

sensor network (WSN) consisting of small-size wireless sensor nodes equipped with radio and one or several sensors, is an attractive and cost-efficient option to build the required measurement system. In this work, we developed a wireless sensor node for greenhouse monitoring by integrating a sensor platform provided SunSPOT by Sun Microsystems with few sensors capable to measure four climate variables. Continuous advancements in wireless technology and miniaturization have made the deployment of sensor networks to monitor various aspects of the environment increasingly flexible.

2. Mobile platform

Mobile robotics is a young field of research. Its roots include many engineering and science disciplines, from mechanical, electrical and electronics engineering to computer, cognitive and social sciences. The Board Of Education is a complete, low-cost development platform equipped with the needed sensors for humidity, temperature, light, etc. As shown in Figure 1, the Boe-Bot is a great tool with which to get started with robotics.

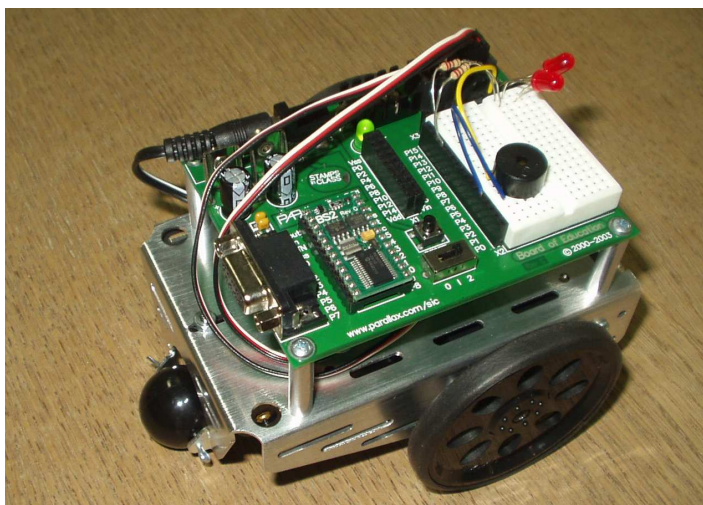


Fig. 1. Assembled Boe-Bot

The SunSPOT WSN module makes it possible for the Boe-Bot robot's BASIC Stamp 2 microcontroller brain to communicate wirelessly with a web based user interface running on a nearby PC. The BASIC Stamp microcontroller runs a small PBASIC program that controls the Boe-Bot robot's servos and optionally monitors sensors while it communicates wirelessly with the web server.

3. Control scheme for mobile robots

A mobile robot needs locomotion mechanisms that enable it to move throughout its known or unknown environment. But there are a large variety of possible ways to move, and so the selection of a robot's approach to locomotion is an important aspect of mobile robot design.

Figure 2, presents the control scheme for mobile robot systems. In the laboratory, there are research robots that can walk, jump, run, slide, skate, swim, fly, and, of course, roll. Any of these activities has its own control algorithm (Gy. Mester, 2009).

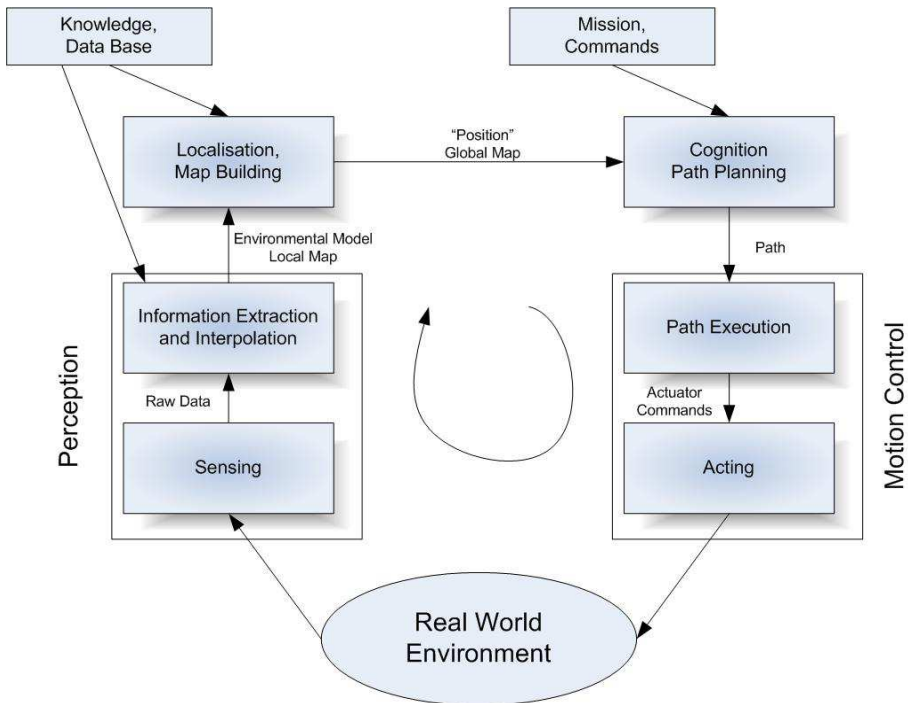


Fig. 2. Reference control scheme for mobile robot systems

Locomotion is the complement of manipulation. In manipulation, the robot arm is fixed but moves objects in the workspace by imparting force to them. In locomotion, the environment is fixed and the robot moves by imparting force to the environment. In both cases, the scientific basis is the study of actuators that generate interaction forces, and mechanisms that implement desired kinematical and dynamic properties. The wheel has been by far the most popular mechanism in mobile robotics and in man-made vehicles in general. It can achieve very good efficiencies, and does so with a relatively simple mechanical implementation. On Figure 3, the kinematics of the mobile robot is depicted. In addition, balance is not usually a research problem in wheeled robot designs, because wheeled robots are almost always designed so that all wheels are in ground contact at all times (Gy. Mester, 2009).

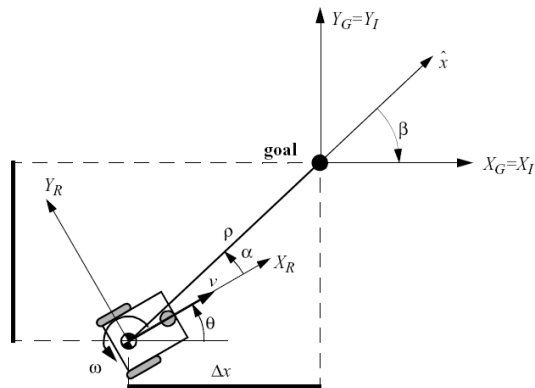


Fig. 3. Robot kinematics and its frames of interests

Thus, three wheels are sufficient to guarantee stable balance, although, as we shall see below, two-wheeled robots can also be stable (R. Siegwart, 2004). When more than three wheels are used, a suspension system is required to allow all wheels to maintain ground contact when the robot encounters uneven terrain. Motion control might not be an easy task for this kind of systems. However, it has been studied by various research groups, and some adequate solutions for motion control of a mobile robot system are available (Gy. Mester, 2009).

4. Using Potential Fields method for navigation

A potential field consists of two imaginary fields (*attractive potential* and *repulsive potential*) and used to avoid a collision with unexpected obstacle while moving in a predetermined path. The Attractive Potential forces the robot to move through a predetermined path and the Repulsive Field, assumed to be generated by obstacles, forces the robot to move a different way to avoid the collision (O. Khatib, 1986). The Artificial Potential Field approach is a local path planner method that was introduced by Khatib. This method defines obstacles as repelling force sources, and goals as attracting force sources. The path is then influenced by the composition of the two forces, which produces a robot motion that moves away from obstacles while moving towards the target goal. The approach is mathematically simple and is able to produce real-time acceptable results for collision avoidance even in dynamic environments. The most known limitation of this approach is the local minima, which refers to locations that trap the robot and prevent it from reaching the target goal location. This main problem has been addressed by many different techniques that try to solve or at least minimize its impact (O. Khatib, 1985).

4.1 Attractive Potential Field

The attractive potential field corresponds to the component responsible for the potentials that attract the robot towards the target goal position. At all locations in the environment the action vector will point to the target goal.

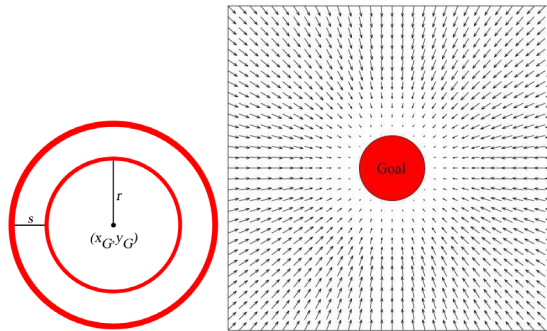


Fig. 4. Attractive potential field action vectors pointing to the goal and goal representation (M. Goodrich, 2002)

Usually, the action vector is found by applying a scalar potential field function to the robot's position and then calculating the gradient of that function.

$$\nabla = [\nabla x, \nabla y] = \left[\frac{\partial U}{\partial x}, \frac{\partial U}{\partial y} \right] \quad (1)$$

After defining: (M. Goodrich, 2002)

- $[x_G, y_G]$ as the position of the goal;
- r as the radius of the goal;
- $[x_R, y_R]$ as the position of the robot;
- s as the size of the goal's area of influence;
- α as the strength of the attractive field ($\alpha > 0$)

We can compute ∇x and ∇y using the following steps:

1. Find the distance d between the goal and the robot:

$$d = \sqrt{(x_G - x_R)^2 + (y_R - y_G)^2} \quad (2)$$

2. Find the angle θ between the robot and the goal:

$$\theta = \tan^{-1} \left(\frac{y_G - y_R}{x_G - x_R} \right) \quad (3)$$

3. Set ∇x and ∇y according to the rules:

If $d < r$ then $\nabla x = \nabla y = 0$

$$\text{If } r \leq d \leq s + r \text{ then } \begin{cases} \nabla x = \alpha(d - r) \cos(\theta) \\ \nabla y = \alpha(d - r) \sin(\theta) \end{cases} \quad (4)$$

$$\text{If } d > s + r \text{ then } \begin{cases} \nabla x = \alpha s \cos(\theta) \\ \nabla y = \alpha s \sin(\theta) \end{cases}$$

The last step presents three simple rules that characterize three different behaviors for the robot according to its relative position towards the goal:

- In the first rule of step 3, $d < r$ means that the robot is in the goal area. In this case, no forces act and ∇x and ∇y are set to zero.
- In the second rule, $r \leq d \leq s + r$ means that the robot is inside the area of influence of the goal. The action vector is set using α , d and s .
- In the third and last rule, $d > s + r$ means that the robot is outside the goal area and also outside its area of influence. The action vector is set with s and α thus reaching higher values.

4.2 Repulsive Potential Field

The repulsive potential field is the component that is responsible for forcing the robot to stay away from the obstacles it encounters on its path. All repulsive action vectors point away from the obstacle surface driving the robot away from the obstacle.

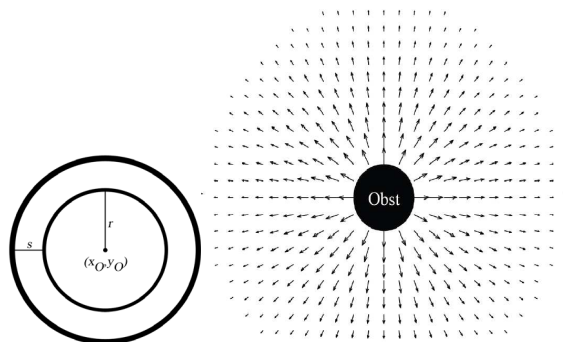


Fig. 5. Repulsive potential field action vectors pointing away from the obstacle and obstacle representation (M. Goodrich, 2002)

Similarly to the Attractive Potential, we calculate the repulsive action vector.

After defining: (M. Goodrich, 2002)

- $[x_O, y_O]$ as the position of the obstacle;
- r as the radius of the obstacle;
- $[x_R, y_R]$ as the position of the robot;
- s as the size of the obstacle's area of influence;
- β as the strength of the repulsive field ($\beta > 0$)

We can compute ∇x and ∇y using the following steps:

1. Find the distance d between the obstacle and the robot:

$$d = \sqrt{(x_O - x_R)^2 + (y_O - y_R)^2} \quad (5)$$

2. Find the angle θ between the robot and the obstacle:

$$\theta = \tan^{-1} \left(\frac{y_O - y_R}{x_O - x_R} \right) \quad (6)$$

3. Set ∇x and ∇y according to the rules:

$$\begin{aligned} \text{If } d < r \text{ then } & \begin{cases} \nabla x = -\text{sign}(\cos(\theta))\infty \\ \nabla y = -\text{sign}(\sin(\theta))\infty \end{cases} \\ \text{If } r \leq d \leq s + r \text{ then } & \begin{cases} \nabla x = -\beta(s + r - d)\cos(\theta) \\ \nabla y = -\beta(s + r - d)\sin(\theta) \end{cases} \end{aligned} \quad (7)$$

If $d > s + r$ then $\nabla x = \nabla y = 0$

Similar to the attractive potential rules, these rules are also simple and characterize three different behaviors for the robot according to its position relative to the obstacle. It is important to notice that all action vectors need to point away from the obstacle, hence the need to use negative values (M. Goodrich, 2002).

- In the first rule of step 3, the robot is within the radius of the obstacle, so the action vector needs to be infinite, expressing the need to escape from the robot.

- In the second rule, where the robot is outside the obstacle's radius but inside its area of influence, the action vector is set to a high value in order to express the need to escape the current location.
- In the third rule, where the robot is outside the area of influence of the obstacle, the action vector is set to zero, meaning that no repulsive forces are acting on the robot (M. Goodrich, 2002).

Since the repulsive force only acts when the robot is inside the area of influence of the obstacle, the value of S must be carefully chosen. A small value for S can cause trajectory problems by causing abrupt changes on the path and some constraints on the speed of the robot. A large value for s may cause also problems on the robot's movement since it can constrain movement in small places where the robot could pass.

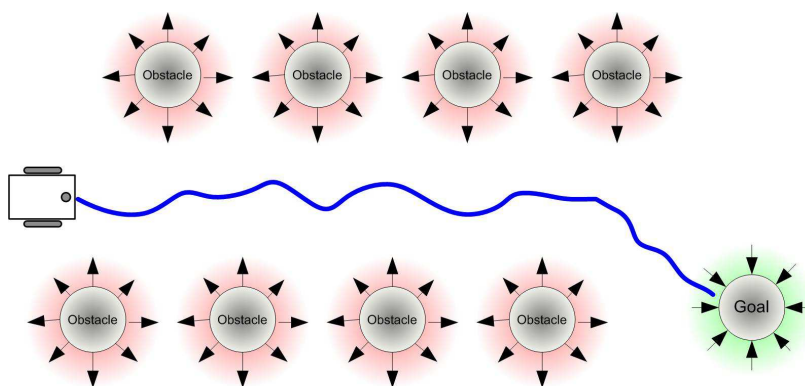


Fig. 6. Potential Fields simulation

The repulsive force has the objective of repelling the robot only if it is close to an obstacle and its velocity points towards that obstacle (M. Goodrich, 2002).

4. WSN and Event-Based System for Greenhouse Climate Control

A wireless sensor network (WSN) is a computer network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations (Sun Microsystems, 2002). The development of wireless sensor networks was originally motivated by military applications such as battlefield surveillance. Figure 7, presents the sensor node architecture. However, wireless sensor networks are now used in many civilian application areas, including environment and habitat monitoring, healthcare applications, home automation, and traffic control.

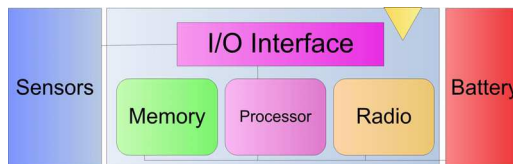


Fig. 7. Sensor Node Architecture

In addition to one or more sensors, each node in a sensor network is typically equipped with a radio transceiver or other wireless communications device, a small microcontroller, and an energy source, usually a battery. Figure 8, shows the typical wireless sensor network.

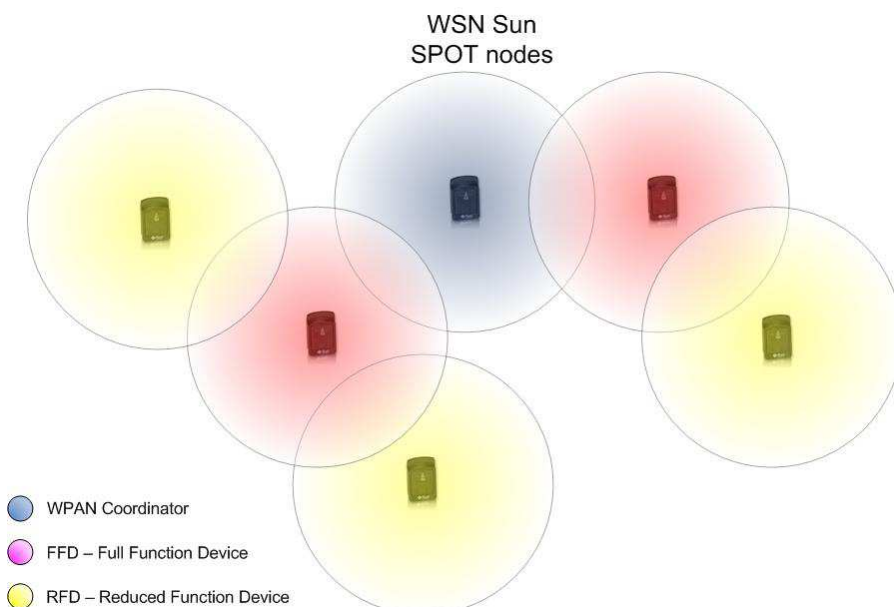


Fig. 8. Typical wireless sensor network (WSN)

The size a single sensor node can vary from shoebox-sized nodes down to devices the size of grain of dust. The cost of sensor nodes is similarly variable, ranging from hundreds of dollars to a few cents, depending on the size of the sensor network and the complexity required of individual sensor nodes (Sun Microsystems, 2005). Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and bandwidth. In computer science, wireless sensor networks are an active research area with numerous workshops and conferences arranged each year (S. Scaglia, 2008). As commented above, this paper is devoted to analyzing diurnal and nocturnal temperature control with natural ventilation and heating systems, and humidity control as a secondary control objective. Under diurnal conditions, the controlled variable is the inside temperature and the control signal is the vent opening. The use of natural ventilation produces an exchange between the inside and outside air, usually provoking a decrease in the inside temperature of the greenhouse. The controller must calculate the

necessary vent opening to reach the desired setpoint. The commonest controller used is a gain scheduling PI scheme where the controller parameters are changed based on some disturbances: outside temperature and wind speed. In the case of nocturnal temperature control, forced-air heaters are used to increase the inside temperature and an on/off control with dead/zone was selected as heating controller.

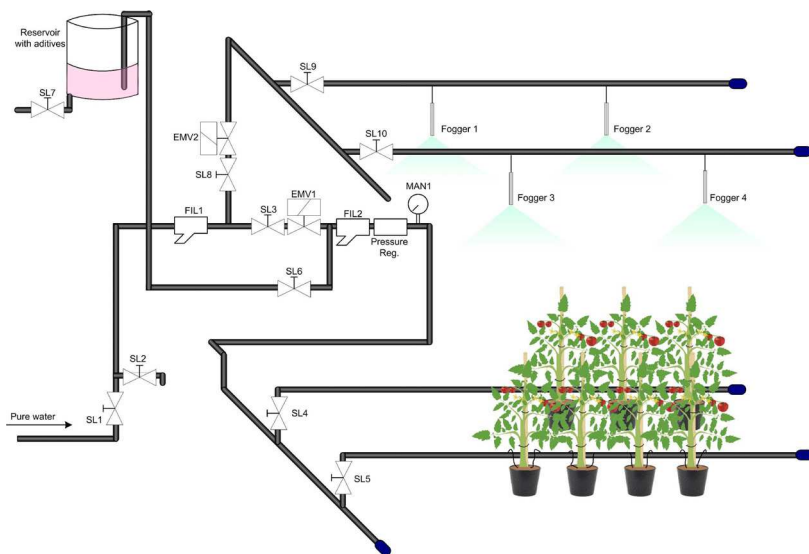


Fig. 9. Humidity control

Climate monitoring is vitally important to the operation in greenhouses and the quality of the collected information has a great influence on the precision and accuracy of control results. Currently, the agro-alimentary market field incorporates diverse data acquisition techniques (S. Scaglia, 2008). Normally, the type of acquisition system is chosen to be optimal for the control algorithm to be used. For traditional climate monitoring and control systems, all sensors are distributed through the greenhouse and connected to the device performing the control tasks. These equipments use time-based data sampling techniques as a consequence of using time-based controllers. Figure 10, presents the temperature controller.

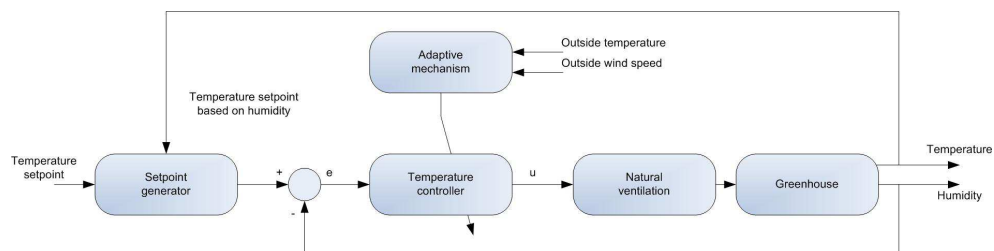


Fig. 10. Temperature controller

Nowadays, commercial systems present more flexibility in the implementation of control algorithms and sampling techniques, especially WSN, where each node of the network can be programmed with a different sampling algorithm or local control algorithm with the main goal of optimizing the overall performance.

4.1 Description of the Sun SPOT WSN module

One of the most popular technologies in the WSN area is Sun SPOT (Small Programmable Object Technology). It contains 32-bit ARM9 CPU, 512K memory, 2 Mb flash storage and wireless networking is based on ChipCon CC2420 following the 802.15.4 standard with integrated antenna and operates in the 2.4GHz to 2.4835GHz ISM unlicensed bands. The IC contains a 2.4GHz RF transmitter/receiver with digital direct sequence spread spectrum (DSSS) baseband modem with MAC support.

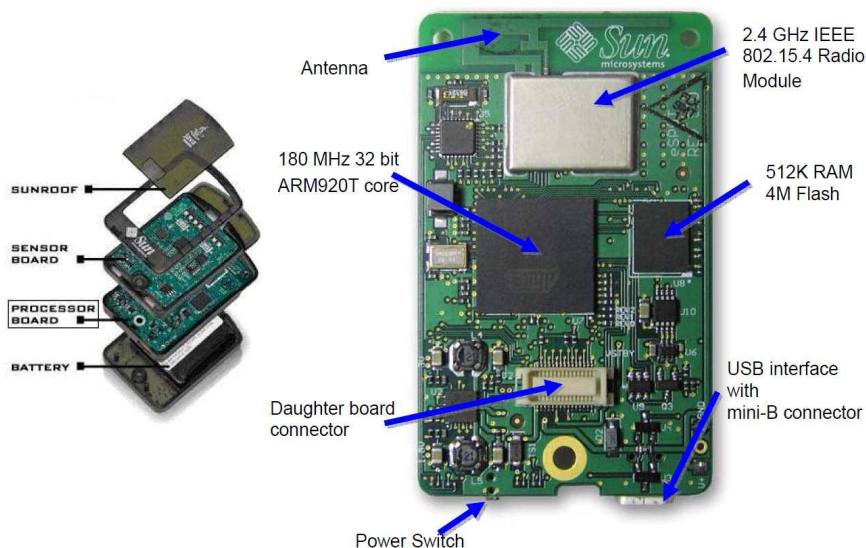


Fig. 11. Sun SPOT processor board

The sensor board integrates multiple sensors, monitoring LED and interactive switches into one board. All the facilities of this board are programmable in Java. The Sun SPOT SDK comes with two important tools for managing the software on your SPOTs: SPOTManager and SPOTWorld. The SPOTManager is a tool for managing the Sun SPOT SDK software. You can use it to download from the Internet both new and old versions of the Sun SPOT SDK. You can use it to make one or another SDK the active SDK on your host workstation, and you can use it to download system software to your Sun SPOTs (Sun Microsystems, 2005).

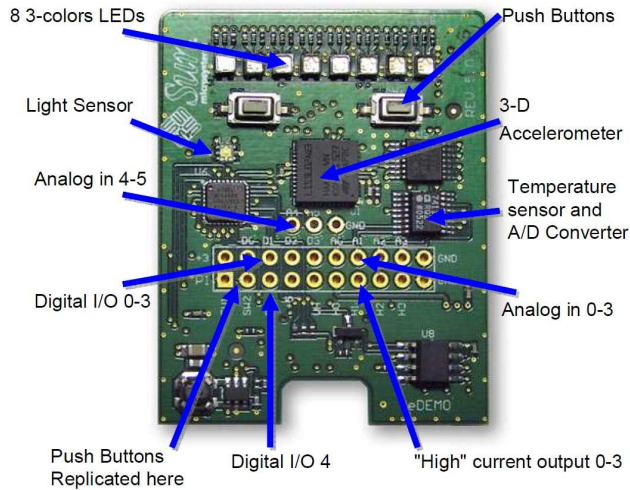


Fig. 12. Sun SPOT sensor board

The facilities of the sensor board are:

- One 2G/6G 3-axis accelerometer
- One temperature sensor
- One light sensor
- Two 8-bit tri-color LEDs
- 6 analog inputs
- Two momentary switches
- 5 general purpose I/O pins

The internal battery is a 3.7V rechargeable lithium-ion prismatic cell. The battery has internal protection circuit to guard against over discharge, under voltage and overcharge conditions. The battery can be charged from either the USB type mini-B device connector or from an external source with a 5V power supply.

4.2 Wireless radio

The wireless network communications uses an integrated radio transceiver, the TI CC2420 (formerly ChipCon). The CC2420 is IEEE 802.15.4 compliant device and operates in the 2.4GHz to 2.4835GHz ISM unlicensed bands. Regulations for these bands are covered by FCC CFR47 part 15 (USA), ETSI EN 300 328 and EN 300 440 class 2 device (Europe) and ARIB STD-T66 (Japan).

The IC contains a 2.4GHz RF transmitter/receiver with digital direct sequence spread spectrum (DSSS) baseband modem with MAC support. Other features include separate TX and RX 128 byte FIFOs, AES encryption (currently not supported), received signal strength indication (RSSI) with 100dB sensitivity and transmit output power setting from -24dBm to 0dBm. Effective bit rate is 250kbps and chip rate is 2000kChips/s. Receive sensitivity is -

90dBm. The digital control and data communications with the CC2420 use PIO port bits and the SPI channel. The CC2420 is a slave SPI bidirectional device addressed when RF_CS (PCS2) is asserted active low. PIO ports reset the CC2420 (RF_RST), power it down (RF_PWDOWN), or check the status of the receive FIFO (FIFO and FIFOP), clear channel assessment (CCA) and start of frame (SFD). There are 33 configuration and status registers, 15 command registers and two 8-bit registers for the separate transmit and receive FIFOs. The first byte sent to the CC2420 is the address made up of 6-bit address, RAM/Register select (Bit 7) and Read/Write select (Bit 6). Following bytes are data read from or written to the CC2420.

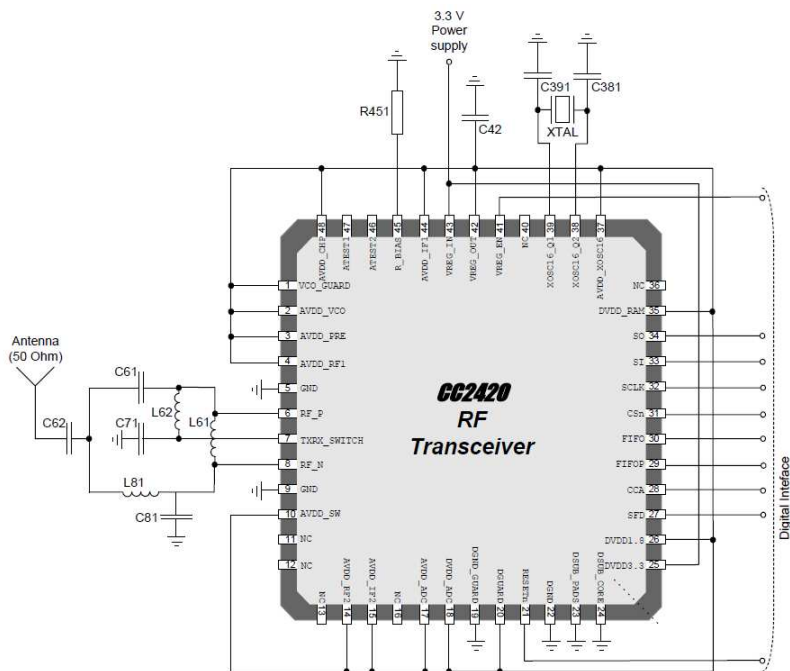


Fig. 13. Typical application circuit for CC2420

The CC2420 is housed in a 48pin quad leadless package (QLP or QFN) that is 7mm square. It is powered with +3.3V Vcc supply. The CC2420 has an internal 1.8V low drop out regulator for powering the internal RF and analog circuitry. It consumes 20mA during receive operation and 18mA for 0dBm transmit. The frequency generation uses an accurate 16MHz crystal with $\pm 10\text{ppm}$ accuracy, $\pm 10\text{ppm}$ stability and $\pm 1\text{ppm}$ aging. The entire RF section is enclosed in an upper and lower RF shield and has modular FCC approval.

4.3 I/O pin Manipulation of the SunSPOT module

The SunSPOT's sensor board has an Atmega88 and operates the 8 tricolor LED's, the accelerometer configuration, and the following pins on the I/O header: I/O pins D0 through D3 can be set as either an output or input.

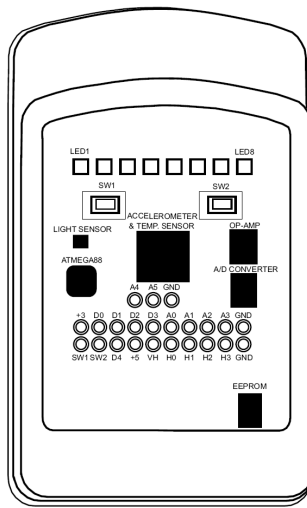


Fig. 14. The SunSPOT's sensor board component location

The high current driver pins, H0 to H3, can only be used as an output. If configured as an output, the pin may be set hi, low, or toggled to its opposite state.

Developing environment for Sun SPOTs

The Java programming language is a general-purpose concurrent class-based object-oriented programming language, specifically designed to have as few implementation dependencies as possible. It allows application developers to write a program once and then be able to run it everywhere on the Internet. Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. The original and reference implementation Java compilers, virtual machines, and class libraries were developed by Sun from 1995. As of May 2007, in compliance with the specifications of the Java Community Process, Sun made available most of their Java technologies as free software under the GNU General Public License. One characteristic of Java is portability, which means that computer programs written in the Java language must run similarly on any supported hardware/operating-system platform. One should be able to write a program once, compile it once, and run it anywhere. This is achieved by compiling the Java language code, not to machine code but to Java bytecode – instructions analogous to machine code but intended to be interpreted by a virtual machine (VM) written specifically for the host hardware. End-users commonly use a Java Runtime Environment (JRE) installed on their own machine for standalone Java applications, or in a Web browser for Java applets. Standardized libraries provide a generic way to access host specific features such as graphics, threading and networking. In some JVM versions, bytecode can be compiled to native code, either before or during program execution,

resulting in faster execution (J. Gosling, 2005). The most popular developing environment for Java is Netbeans IDE.

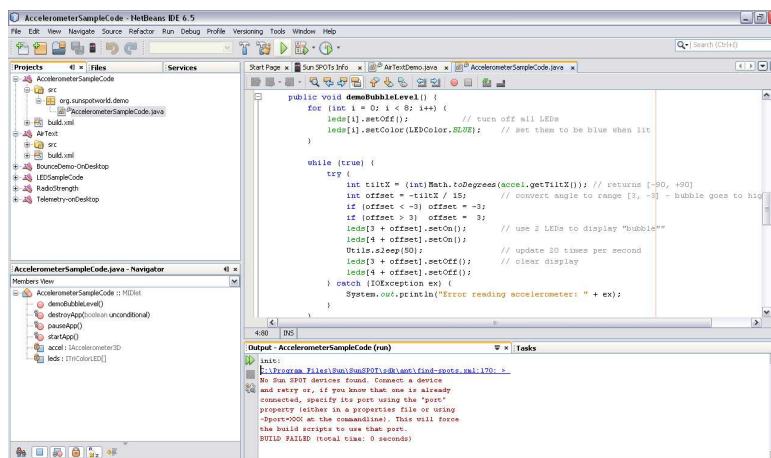


Fig. 15. NetBeans IDE 6.5 – Sun SPOT SDK

A major benefit of using bytecode is porting. However, the overhead of interpretation means that interpreted programs almost always run more slowly than programs compiled to native executables would, and Java suffered a reputation for poor performance. This gap has been narrowed by a number of optimization techniques introduced in the more recent JVM implementations. One such technique, known as just-in-time (JIT) compilation, translates Java bytecode into native code the first time that code is executed, then caches it. This results in a program that starts and executes faster than pure interpreted code can, at the cost of introducing occasional compilation overhead during execution. More sophisticated VMs also use dynamic recompilation, in which the VM analyzes the behavior of the running program and selectively recompiles and optimizes parts of the program. Dynamic recompilation can achieve optimizations superior to static compilation because the dynamic compiler can base optimizations on knowledge about the runtime environment and the set of loaded classes, and can identify hot spots - parts of the program, often inner loops, that take up the most execution time. JIT compilation and dynamic recompilation allow Java programs to approach the speed of native code without losing portability.

4.4 The Sun SPOT emulator

Solarium includes an emulator capable of running a Sun SPOT application on your desktop computer. This allows for testing a program before deploying it to a real SPOT, or if a real SPOT is not available. Instead of a physical sensor board, Solarium displays a virtual SPOT with a control panel where you can set any of the potential sensor inputs (e.g. light level, temperature, digital pin inputs, analog input voltages, and accelerometer values). Your application can control the LEDs' color that is displayed in the virtual SPOT image, just like it would a real SPOT. You can click with the mouse on the push button switches in the virtual SPOT image to press and release the switches. Receiving and sending via the radio is also supported. Each virtual SPOT is assigned its own address and can broadcast or unicast

to the other virtual SPOTs. If a shared base station is available a virtual SPOT can also interact over the radio with real SPOTs.

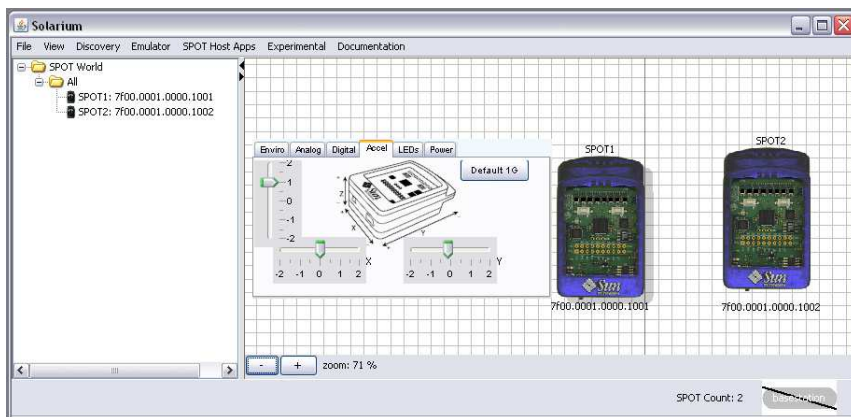


Fig. 16. The Sun SPOT Emulator

Virtual SPOTs can communicate with each other by opening radio connections, both broadcast and point-to-point. Instead of using an actual radio these connections take place over regular and multicast sockets. When a base station SPOT is connected to the host computer and a shared base station is running, virtual SPOTs can also use it to communicate with real SPOTs using the base station's radio. The advantage of using a shared base station is that multiple host applications can then all access the radio. One disadvantage is that communication from a host application to a target SPOT takes two radio hops, in contrast to the one hop needed with a dedicated base station. Another disadvantage is that run-time manipulation of the base station SPOT's radio channel, pan id or output power is not currently possible. Each virtual SPOT has its own Squawk VM running in a separate process on the host computer. Each Squawk VM contains a complete host-side radio stack as part of the SPOT library, which allows the SPOT application to communicate with other SPOT applications running on the host computer, such as other virtual SPOTs, using sockets or real SPOTs via radio if a shared base station is running. The current Solarium implementation is primarily an emulator since it actually runs a SPOT application in a Squawk VM, just like the VM on a real SPOT. Likewise radio interaction between virtual SPOTs is emulated with data sent via packets and streams from one (virtual) SPOT to another. Only the SPOT's interaction with the environment is simulated using a simple model where the user needs to explicitly set the current sensor values. Future versions may incorporate more simulation of SPOT properties like battery level or radio range.

5. Solution

Building and programming a robot is a combination of mechanics, electronics, and problem solving. What you're about to learn while doing the activities and projects in this text will be relevant to "real world" applications that use robotic control, the only difference being the size and sophistication. Robotics has come a long way, especially for mobile robots. In the past, mobile robots were controlled by heavy, large, and expensive computer systems that could not

be carried and had to be linked via cable or wireless devices. As shown in Figure 8, the mobile measuring station is navigating inside the greenhouse. Today, however, we can build small mobile robots with numerous actuators and sensors that are controlled by inexpensive, small, and light embedded computer systems that are carried on-board the robot.

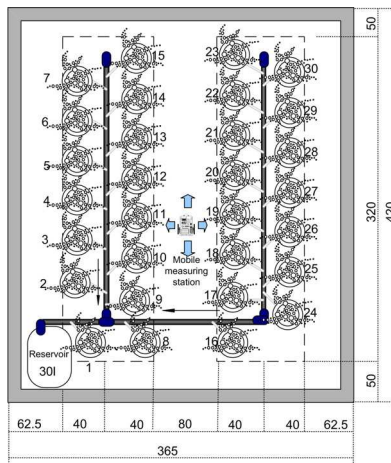


Fig. 17. Greenhouse top view with the mobile measuring station

The mechanical principles, example program listings, and circuits you will use are very similar to, and sometimes the same as, industrial applications developed by engineers. In this project we have used SunSPOT-s to achieve remote control over a Boe-Bot. For this project we have used 2 SunSPOT-s from the kit (free range and base station module) as depicted on Figure 18. SunSPOT's wireless protocol is Zigbee based protocol (I. Matijevics, 2008).

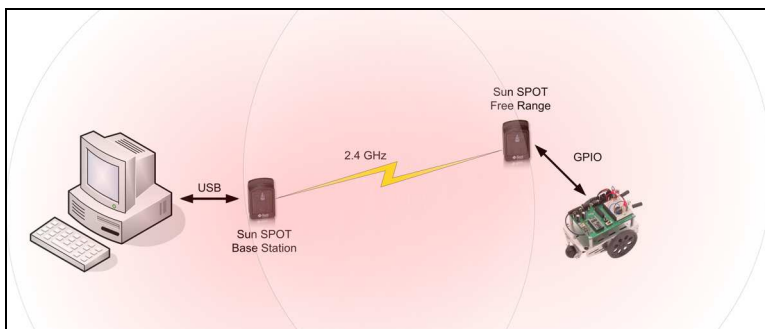


Fig. 18. Connection of the system

The Hardware basically centers around Sun SPOT and DC Motors controlled by Basic Stamp. The Sun SPOT base station will send data to Sun SPOT on the mobile measuring station which will drive the Basic Stamp controller to DC IO pins (J. Simon, 2009). The microcontroller will drive the Motors which will run the measuring station. Figure 19, shows the testing phase of the mobile measuring station.

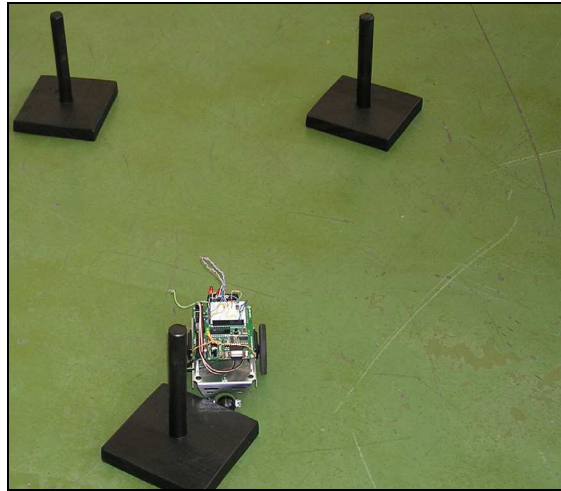


Fig. 19. Boe-bot with SunSPOT mounted

6. Experimental results

The applications for WSNs are many and varied. They are used in commercial and industrial applications to monitor data that would be difficult or expensive to monitor using wired sensors. They could be deployed in wilderness areas, where they would remain for many years (monitoring some environmental variable) without the need to recharge/replace their power supplies. They could form a perimeter about a property and monitor the progression of intruders (passing information from one node to the next). There are a many uses for WSNs (I. Matijevics, 2009).



Fig. 20. Crops in greenhouse

Typical applications of WSNs include monitoring, tracking, and controlling. Some of the specific applications are habitat monitoring, object tracking, nuclear reactor controlling, fire detection, traffic monitoring, etc. In a typical application, a WSN is scattered in a region where it is meant to collect data through its sensor node. Figure 23, shows the complete control system of the greenhouse. The WSN-based controller has allowed a considerable decrease in the number of changes in the control action and made possible a study of the compromise between quantity of transmission and control performance.



Fig. 21. Capsicum

Motion control of mobile robots is a very important research field today, because mobile robots are a very interesting subject both in scientific research and practical applications. In this paper the object of the remote control is the Boe-Bot. The vehicle has two driving wheels and the angular velocities of the two wheels are independently controlled (A. Pawlowski, 2009). When the vehicle is moving towards the target and the sensors detect an obstacle, an avoiding strategy is necessary. The host system connects to the mobile robot with the SunSPOT module. A remote control program has been implemented as shown on Figure 22.

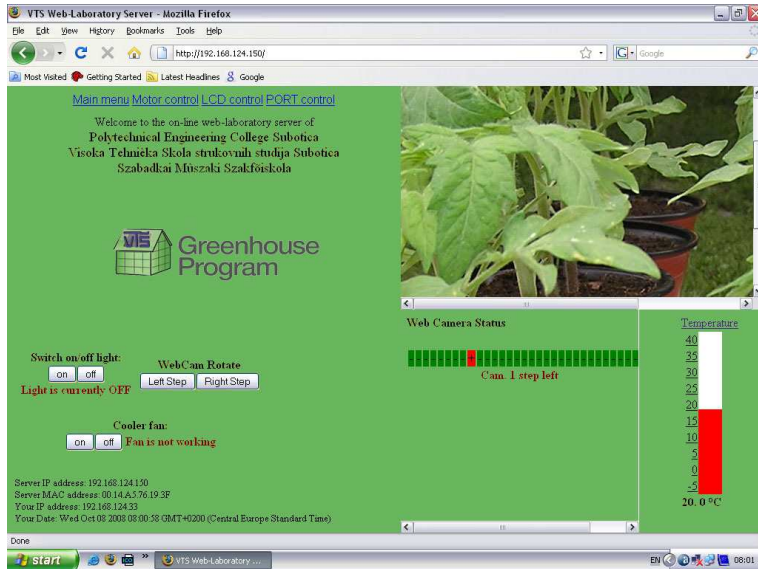


Fig. 22. Control surface of the system

The limit of the level crossing sampling has presented a great influence on the event based control performance where, for the greenhouse climate control problem, the system has provided promising results.

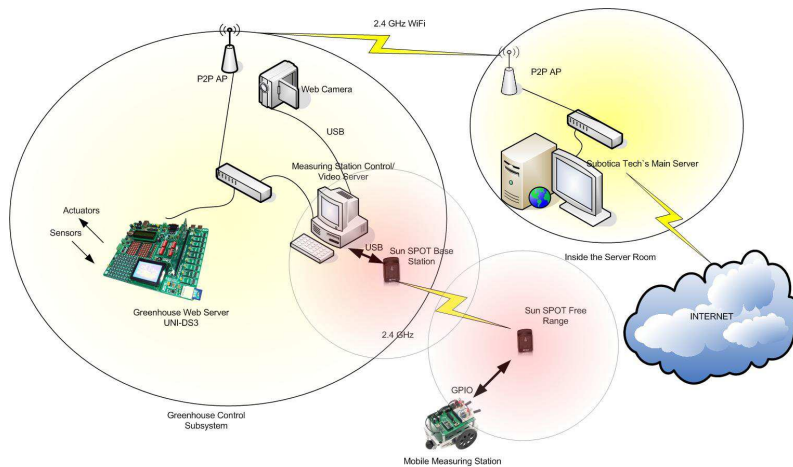


Fig. 23. Greenhouse control system

The code snippet below gives an example for testing the communication devices in broadcast mode as we can see on Figure 24. It is written in Java and runs on SunSPOT modules. Each SPOT is assigned its own address and can broadcast or unicast to the other SPOTs. This code is implemented for testing purposes only.

```

protected void startApp() throws MIDletStateChangeException {
    System.out.println("Broadcast Counter MIDlet");
    //showColor(color);
    //switches[0].addISwitchListener(this);
    //switches[1].addISwitchListener(this);
    try {
        tx
        (RadiogramConnection)Connector.open("radiogram://broadcast:123");
        xdg = (Radiogram)tx.newDatagram(20); //transmitting the
radiogram
        RadiogramConnection rx
        (RadiogramConnection)Connector.open("radiogram://:123");
        Radiogram rdg = (Radiogram)rx.newDatagram(20);
        //outs[0].setHigh();
        while (true) {
            try {
                rx.receive(rdg);
                int cmd = rdg.readInt();
                //int newCount = rdg.readInt();
                //int newColor = rdg.readInt();
                /*if (cmd == CHANGE_COLOR) {
                    System.out.println("Received packet from " +
rdg.getAddress());
                    //showColor(newColor);
                } else {
                    //showCount(newCount, newColor);
                }*/
                switch (cmd){
                    case 0: outs[demo.H0].setLow();
outs[demo.H1].setLow(); leds[0].setRGB(200, 0, 0); leds[0].setOn();
leds[1].setOff();leds[2].setOff();leds[3].setOff(); break;
                    case 4: outs[demo.H0].setHigh();
outs[demo.H1].setLow(); leds[1].setRGB(200, 0, 0); leds[1].setOn();
leds[0].setOff();leds[2].setOff();leds[3].setOff(); break;
                    case 3: outs[demo.H0].setLow();
outs[demo.H1].setHigh(); leds[2].setRGB(200, 0, 0); leds[2].setOn();
leds[1].setOff();leds[0].setOff();leds[3].setOff(); break;
                    case 1: outs[demo.H0].setHigh();
outs[demo.H1].setHigh(); leds[3].setRGB(200, 0, 0); leds[3].setOn();
leds[1].setOff();leds[2].setOff();leds[0].setOff(); break;
                    //setting up the diagnostic leds
                    default: leds[4].setRGB(200, 0, 0);
leds[4].setOn(); break;
                }
            } catch (IOException ex) {
                System.out.println("Error receiving packet: " + ex);
                ex.printStackTrace(); // Error detection
            }
        }
    } catch (IOException ex) {
        System.out.println("Error opening connections: " + ex);
        ex.printStackTrace(); // Error detection
    }
}

```

Fig. 24. Sending broadcast packets via wsn from base station

The Sun SPOT is a Java programmable embedded device designed for flexibility. The basic unit includes accelerometer, temperature and light sensors, radio transmitter, eight multicolored LEDs, 2 push-button control switches, 5 digital I/O pins, 6 analog inputs, 4 digital outputs, and a rechargeable battery. Java implementation and programming the Sun SPOT is surprisingly easy. Experimental testing has demonstrated the validity of our approach.

7. Comparison of the fruit production

Tomatoes are a warm season vegetable crop. They grow best under conditions of high light and warm temperatures. Low light in a fall or winter greenhouse, when it is less than 15% of summer light levels, greatly reduces fruit yield when heating costs are highest. For this reason, it is difficult to recommend that a greenhouse operator should grow and harvest fruit from December 15 to February 15. Based on few years of experience, tomato production is most successful in the spring. Excellent light, moderate heating costs and good prices annually demonstrate this is the best time for greenhouse tomato production. Tomato plants grow best when the night temperature is maintained at 16 - 18 °C. Temperatures below 16 °C will prevent normal pollination and fruit development. In warm or hot outdoor conditions, tomato greenhouses must be ventilated to keep temperatures below 35 °C. High temperatures not only effect the leaves and fruit, but increased soil temperatures also reduce root growth. Table 1, gives an overview of effectiveness of the control system.

Tested plants	Average weight of fruit (with WSN control)	Average weight of fruit (without WSN control)	Average number of fruit per plant (with WSN control)	Average number of fruit per plant (without WSN control)
Tomato	210 g	180 g	17	11
Capsicum	135 g	110 g	15	12
Cucumber	70 g	60 g	13	10

Table 1. The average total weight and number of fruit harvested

Success in greenhouse plants depends completely on fruit yield. Yields of 20 - 25 % gain per plant are very good for annual costs.

8. Conclusion

The system and its implementation have been successful, however there are still possibilities for further development. The first cycle of plant development has just passed, and it has provided numerous valuable data. For the next cycle will better conditions will be provided, with a more experienced staff. With further developments the application of professional industrial electronics will also have to be taken into consideration, which would significantly decrease possible problems. One of the research areas inside the Greenhouse program is wirelessly controlled mobile measuring station. Traditionally research into autonomous robotics has been performed on robotics platforms that cost tens of thousands of dollars. As an alternative, several research groups have developed low-cost robots that

are controlled by a SunSPOT node running Java VM. One of the goals of this project is to develop algorithms for coordination and navigation inside the greenhouse.

9. Acknowledgement

This research was partially supported by the TÁMOP 4.2.1/B-09/1/KONV-2010-0005 program of the Hungarian National Development Agency.

10. References

- A. Pawlowski, J. Luis Guzman, F. Rodríguez, M. Berenguel, J. Sánchez and S. Dormido, (2009) "Simulation of Greenhouse Climate Monitoring and Control with Wireless Sensor Network and Event-Based Control" Proceedings of the Conference
- Gy. Mester, (2009) „Wireless Sensor-based Control of Mobile Robot Motion“, Proceeding of the IEEE SISO 2009, pp 81-84, Subotica, Serbia
- Gy. Mester, (2009) „Intelligent Wheeled Mobile Robot Navigation“, Proceedings of the Conference Európai Kihívások V, pp. 1-5, SZTE, Szeged, Hungary
- I. Matijevics, J. Simon, (2008) „Advantages of Remote Greenhouse Laboratory for Distant Monitoring“, Proceedings of the Conference ICoSTAF 2008, pp 1-5, Szeged, Hungary
- I. Matijevics, J. Simon, (2009) „Comparison of various wireless sensor networks and their implementation“, Proceedings of the Conference SIP 2009, pp 1-3, Pécs, Hungary
- J. Gosling, (2005) „The Java™ Language Specification Third Edition“
- J. Simon, G. Martinović, (2009) "Web Based Distant Monitoring and Control for Greenhouse Systems Using the Sun SPOT Modules", Proceedings of the Conference SISO 2009, pp. 1-5, Subotica, Serbia
- J. Simon, I. Matijevics, (2009) „Distant Monitoring And Control For Greenhouse Systems Via Internet“, Zbornik radova konferencije Yuinfo 2009, pp. 1-3, Kopaonik, Srbija
- J. Vasu, L. Shahram, (2008) "Comprehensive Study of Routing Management in Wireless Sensor Networks- Part-1"
- L. Gonda, , C. Cugnasca, (2006) "A proposal of greenhouse control using wireless sensor networks" In Proceedings of 4thWorld Congress Conference on Computers in Agriculture and Natural Resources, Orlando, Florida, USA
- M. J. Mataric, (2007) The Robotics Primer. The MIT Press, 1st edition.
- M. A. Goodrich, (2002) Potential fields tutorial.
- O. Khatib, (1986) "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", Int. J. of Robotic Research, Vol.5, No.1, p.60
- O. Khatib, (1985) "The Potential Field Approach and Operational Space Formulation in Robot Control" Proc. Fourth Yale Workshop on Applications of Adaptive Systems Theory, Yale University, New Haven, Connecticut, pp. 208-214.
- P. Kucsera, (2006) „Sensors For Mobile Robot Systems“ , Academic and Applied Research in Military Science, Volume 5, Issue 4, p.645-658,Hungary
- P. Kucsera, (2006) "Industrial Component-based Sample Mobile Robot System" , Acta Polytechnica Hungarica, Volume 4 Issue Number 4, Hungary
- Roland Siegwart and Illah R., (2004) "Introduction to Autonomous Mobile Robots", Nourbakhsh

- Sun Microsystems Inc. , (2007) „Sun™ Small Programmable Object Technology (Sun SPOT)”
Owner’s Manual Release 3.0
- Sun Microsystems Inc. , (2005) „Sun Spot Developer’s Guide”
- Sun Microsystems Inc. , (2005) „Demo Sensor Board Library”
- S. Scaglia, (2008) „The Embedded Internet”
- X. Feng, T. Yu-Chu, S. Yanjun, S. Youxian, (2007) “Wireless Sensor/Actuator Network
Design for Mobile Control Applications. Sensors” Proceedings of the Conference



Wireless Sensor Networks: Application-Centric Design

Edited by Yen Kheng Tan

ISBN 978-953-307-321-7

Hard cover, 492 pages

Publisher InTech

Published online 14, December, 2010

Published in print edition December, 2010

Over the past decade, there has been a prolific increase in the research, development and commercialisation of Wireless Sensor Networks (WSNs) and their associated technologies. WSNs have found application in a vast range of different domains, scenarios and disciplines. These have included healthcare, defence and security, environmental monitoring and building/structural health monitoring. However, as a result of the broad array of pertinent applications, WSN researchers have also realised the application specificity of the domain; it is incredibly difficult, if not impossible, to find an application-independent solution to most WSN problems. Hence, research into WSNs dictates the adoption of an application-centric design process. This book is not intended to be a comprehensive review of all WSN applications and deployments to date. Instead, it is a collection of state-of-the-art research papers discussing current applications and deployment experiences, but also the communication and data processing technologies that are fundamental in further developing solutions to applications. Whilst a common foundation is retained through all chapters, this book contains a broad array of often differing interpretations, configurations and limitations of WSNs, and this highlights the diversity of this ever-changing research area. The chapters have been categorised into three distinct sections: applications and case studies, communication and networking, and information and data processing. The readership of this book is intended to be postgraduate/postdoctoral researchers and professional engineers, though some of the chapters may be of relevance to interested master's level students.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

István Matijevics and Janos Simon (2010). Improving Greenhouse's Automation and Data Acquisition with Mobile Robot Controlled System via Wireless Sensor Network, *Wireless Sensor Networks: Application-Centric Design*, Yen Kheng Tan (Ed.), ISBN: 978-953-307-321-7, InTech, Available from: <http://www.intechopen.com/books/wireless-sensor-networks-application-centric-design/improving-greenhouse-s-automation-and-data-acquisition-with-mobile-robot-controlled-system-via-wirel>

INTECH
open science | open minds

InTech Europe

University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia

InTech China

Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元

Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

Phone: +86-21-62489820
Fax: +86-21-62489821