Firebird V Based Moisture Sensing And Feedback Tweeting Bot

by

Team 8

Karan Harish Punamiya	**133059004**
Lucky Agarwal	**133050019**
Palash	**133050054**
Sudhir Waghmode	**133050063**

Department of Computer Science & Engineering
Indian Institute of Technology, Bombay
2013

Contents

1 Introduction

Greenhouse automation involves automation of the tasks performed for the entire process of plant growth. In the plant lifecycle, an important activity is proper irrigation of soil to ensure that the plants always have proper hydration. This project involves sensing of soil moisture content and taking appropriate actions for ensuring irrigation of soil.

2 Definitions, Acronyms and Abbreviations

- FireBird: A robot indigenously designed at ERTS laboratory, IIT Bombay.
- AVR-libc: Standard C library implementation by AVR Systems
- COTS: Commercial Off The Shelf

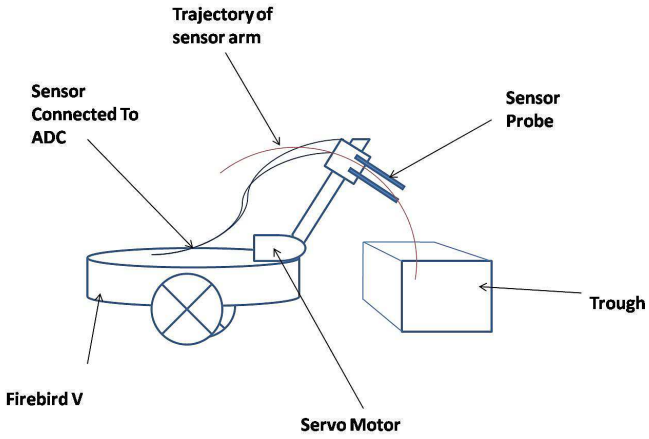

Figure 1: Diagram of our Firebird V with mounted Moisture Sensor

3 Requirements

3.1 Hardware Requirements

1. FireBird V robot
2. Linux machine
3. WiFi access point with Internet connectivity
4. Galvanized iron rods

5. Two ZigBee cards

6. Spark V robot

7. Water pump

8. PVC pipe for dispensing water

3.2 Functional Requirements

1. The user can schedule a sensing operation using the greenhouse interface from remote location.

2. The user specifies exact location to be sensed by specifying the trough number and distance within the trough.

3. The bot performs the sensing operation and reports the sensed values to the server machine.

4. The server machine regulates bot navigation, reads sensed values and takes actions in case the area sensed is dry. The action includes starting the irrigation system and tweeting on the Twitter handle.

5. The server also logs the sensed values for analytics.

3.3 Non Functional Requirements

1. Proper calibration of system for exact moisture value.

2. Response time of one sensing pass.

4 Implementation

The problem statement is divided into following execution steps:

4.1 Interface

Figure 2 shows the user interface for controlling the sensing activity of the bot. It is a web based interface developed using PHP as server side scripting language and HTML with jQuery library for front-end display.
The operations on the interface are as follows:

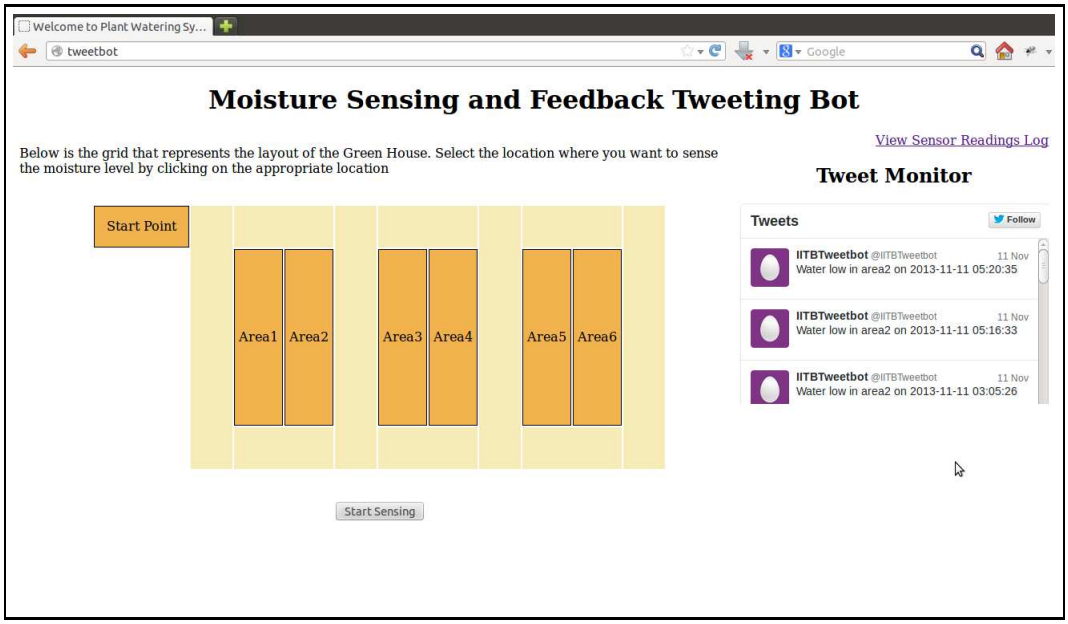

Figure 2: Interactive GUI for specifying sensing schedule

1. The user specifies the locations at which the sensing has to be performed by clicking on the location within the trough.
2. When the user clicks on "Start Sensing", the co-ordinates selected by the user are sent to a backend program that prepares a schedule based on the area locations, and sequentially communicates these locations to the bot.
3. An AJAX process continuously probes whether sensing activity for a location has completed. When completed, it displays the sensed value received on the interface.
4. After completion of the sensing activity, if the sensed values are detected to be below the required levels, the irrigation initiation file is invoked and the incident information is tweeted. The interface also displays the live Twitter feed for the bot's Twitter handle.
5. The logged values of the previously scheduled sensed activities can be accessed via the corresponding link on the screen.

4.2 Bot Guidance System

This will enable the bot to autonomously guide itself to the given coordinates in the greenhouse.The whiteline sensors on the bot are used to keep it on the track at all times and checkpoints are used to judge its curent position.The supporting primitives for this system are given below.

4.3 Moisture Sensing

We have built a cheap soil moisture sensor so that the bot can read the amount of moisture in the soil. The version we are building is very low tech, but it is also very cheap. It consists of two metallic

Primitives	Use
linear_distance_follow_mm()	Moves as per specified distance in linear direction
forward_follow_mm()	Moves in forward direction
isFollowLine()	Checks whether bot is following line
right_upto_follow_line()	Moves right until get follow line
left_upto_follow_line()	Moves left until get follow line
goToRestPosition()	Come back to initial position
goToPreviousPoint()	Come back to previous check point
goToNextPoint()	Goes to next checkpoint
goToLineLane()	Moves corresponding trough
getSensorValue()	Reads sensor value
servo_dig_in()	Sensor arm digs in soil
servo_dig_out()	Sensor arm digs out after sensing the soil
sense()	Take 3 values at one position to remove outlier
senseTrough()	Take 3 values in given location with five degree spacing to remove jitter

Table 1: Bot Guidance System Primitives

probes inserted in the soil a small distance apart. We will report the moisture content of the soil based on the resistance offered by the sensor, more the water lesser the resistance. The calibration of the sensor provides us seperate value of ADC for various moisture states.

Voltage Output(Volts)	**ADC Value(0-255)**	**Moisture State**
2.94-5	150-255	Soil Moisture Adequate
1.37-2.94	70-150	Soil Requires Watering
0-1.37	0-70	Soil Extremely Dry

Table 2: Moisture Sensor Calibration Values

There were many other types of COTS soil moisture sensors such as the Vegetronix VH-400 but we could not use them as they were very costly and our target was to build a low cost sensor.

4.4 Water-Pump Actuation

Our system is capable of identifying dry troughs in the greenhouse and selectively actuating pump irrigation system for only the dry troughs. This prevents water wastage.

4.5 Twitter @IITBTweetbot

We have embeded the latest Twitter 1.2 API to enable us to tweet the status of the greenhouse. The on screen twitter feed gives the latest tweets.

4.6 Data Logging

The system also logs the sensed values in each run to a log file.

5 Design Challenges and Open Issues

The major challenges of the system were to provide accurate results and have quick responses to events that require actions. The various design challenges faced are listed below:

5.1 Sensor Design and Calibration

One of the major challenges in the project was to design a sensor mechanism that can near-accurately measure/quantize the soil moisture level. Also, one of the challenges was to have this sensor design to be low-cost, so as to reduce the overall cost of the mechanism. For accurate measurement of soil moisture, the sensor was calibrated using a series of readings in soils with different moisture levels, and based on sensor value readings, thresholds for mapping voltage value to their associated linguistic terms were determined.

5.2 Bot Navigation

Another concern was to navigate the bot to the location specified in the interface with high accuracy. For handling this, we designed the navigation mechanism to locate the destination location by specifying the distance in centimetres from the nearest checkpoint in the grid, the checkpoints being present at the 2 ends of the trough and one exactly in center. The checkpoint is located using the combination of lane number and checkpoint number within that lane. During the test runs, the bot was able to navigate to locations with error of ± 1 cm with the checkpoints placed every 50 cm.

5.3 Tweeting feedback

Another challenge was to integrating Twitter handling for tweeting about the irrigation events from the server. For using the Twitter handle, the initial RESTful API used was later deprecated by Twitter. The new Twitter API(v1.1) required a developer account to be created on Twitter developers site and registering for using the API service. Also, it required a new OAuth authentication module. For the project, the Twitter API module used is Twitter PHP API available at *https://github.com/J7mbo/twitter-api-php*

6 Conclusion

Irrigation is a major component in the growth of plants. The processes of detecting whether moisture content of soil is low, activating the watering pump et. al. require the user to be physically present at the greenhouse. This project enables automated moisture detection and feedback from remote location using a web interface. This enables the user to schedule a sensing activity from any remote place and receive alerts about the moisture level. Also, the sensing module is linked to the irrigation module, thus reducing the task of manually reacting to the alerts. However, the alerts can be used for detection of any issues with the irrigation module, i.e. if alerts are received multiple times from a particular location in consecutive readings, there might be some fault in the irrigation module.

The individual components of the project such as navigation, tweeting, sensing et. al. are developed as independent modules, and can be re-used in any other project as seems fit. The API used for integrating with Twitter along with the developer handle details are included along with the project source.

7 Design Diagrams

7.1 Statechart for Moisture Sensor

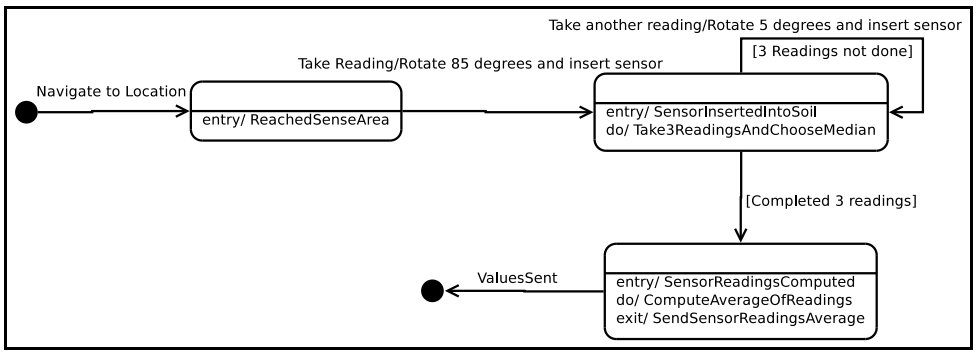

Figure 3: Moisture Sensor

7.2 Statechart of Bot Guidance System

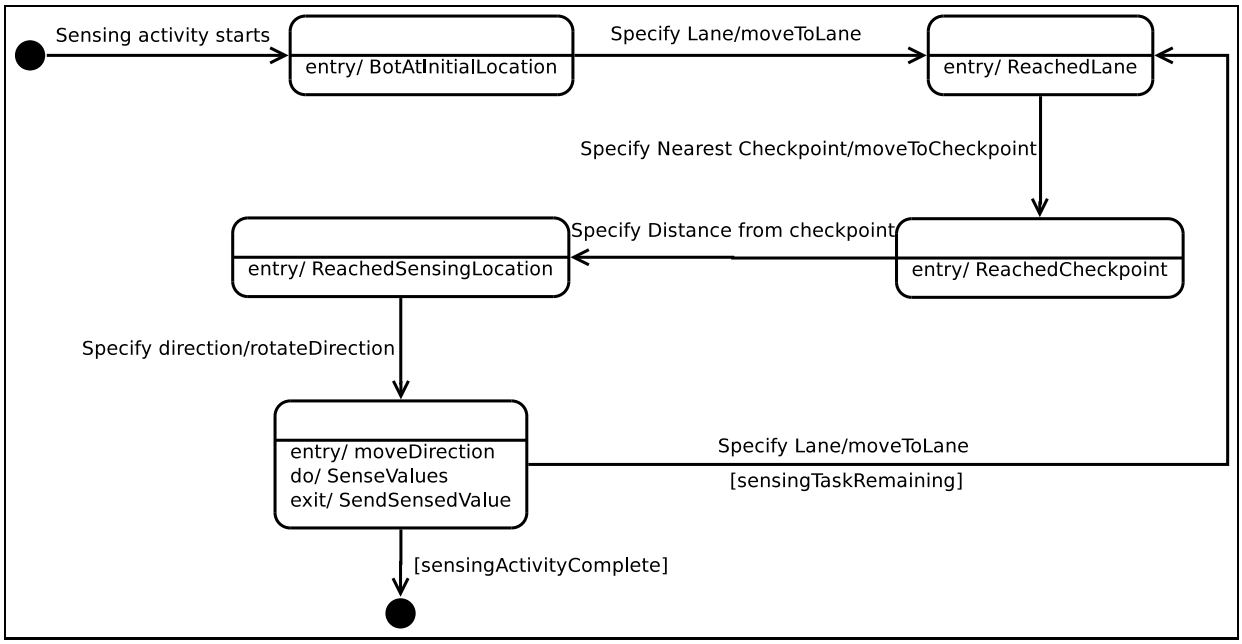

Figure 4: State Chart

8 References

- Garden Bot, *http://www.gardenbot.org*
- Garduino, *www.instructables.com/id/Garduino-Gardening-Arduino/*
- E-yantra website. *http://www.e-yantra.org*.
- Firebird v atmega2560 robotic research platform hardware manual. IIT Bombay & NEX Robotics Pvt. Ltd.
- Firebird v atmega2560 robotic research platform software manual. IIT Bombay & NEX Robotics Pvt. Ltd.