CS-684 Project Report On Autonomous Path Cleaner & Automatic Battery Recharging Bot

Team Name : Team Dynamic
Team No : 13

Team Members :
133050060 Ravi Kumar Yadav
133050062 Om Prakash Swami
133050030 Yogesh Kumar
13305R002 Saransh Sharma

November 14, 2013

Contents

List of Figures

List of Tables

1 Introduction

1.1 Existing System

1. At present there is no existing system that cleans the path of Bot.

2. There is no system that complies Bot Recharging.

1.2 Proposed System

1. The proposed system for **cleaning** , checks the path to be cleaned and then collects the trash in container.

2. The proposed system for **recharging** , checks the current battery status and then takes a decision when to go to the recharging station.

1.3 Final System

The final implemented system can

- Clean the given path and collect the trash in the container.

- Go to the charging point when battery level goes below the minimum threshold level.

1.4 Definitions, Acronyms and Abbrevations

1 FireBird: A robot indigenously designed at ERTS laboratory, IIT Bombay [1].

2 AVR-libc: Standard C library implementation by AVR System [2]

3 Rocky area: It usually refers to naturally occurring solid aggregate of minerals and/or mineraloid

4 GUI: Graphical User Interface (GUI) is the one which helps in depiction of location of objects

2 Problem Statement

Develope an autonomous path cleaning and automatic battery recharging Bot using FireBird V.
Subgoals:-

- designing a automatic cleaning broom.
- designing charging platform.

3 Requirements

3.1 Hardware Requirements

1. Cleaning Bot
   - FireBird V Bot.
   - One DC Motor.
   - One 293D IC.
   - Two Servo Motor.
   - Belt and steel platform.
   - Plastic Container.
   - Cleaning Brush.
   - Medium size wheel.
   - AVR Bootstrap loader
2. Charging Bot
   - FireBird V Bot.
   - Two small metallic strips (for Bot)
   - Two long metallic strips (for Charging Platform)
   - AVR Bootstrap loader

3.2 Software Requirements

- AVR Boot Loader
- AVR Studio
- AVRDude

3.3 Functional Requirements

1. Cleaning Bot
    1. Bot should clean a surface given as grid
    2. Bot should dump trash when path gets covered
    3. Localization without grid (sub goal)
2. Recharging Bot
    1. Bot should get recharged first if battery goes down below a certain level and thereafter it continues from a point where it left cleaning
    2. Decide the shortest path to reach Charging Station

3.4 Non Functional Requirements

1. Cleaning Bot
    1. Field Area to be covered
    2. Size of the grid
    3. Energy Consumption
    4. Low Cost
    5. Less Weight and Noise
2. Recharging Bot
    1. Field Area to be covered
    2. Size of the grid
    3. Mapping Mechanism

4. Low Cost
5. Low Resistance material for metallic strips.
6. Should remember where to go after recharging

3.5 Design Constraints

1. **Cleaning Bot**

   (a) **Grid Arena:** For the system design purpose, the grid based approach is being used. Hence, the 6X6 or 7X7 fixed size grid should be used. Restricting the grid size may be helful for the rigerous testing of the system. Also, the size of the grid to be covered should be much enough for which Bot's battery can last while traversal.

   (b) **Black Lines:** For the working of bot, it has to traverse the grid. hence it need some direction guidelines. Here, black lines of the grid serves this purpose. As the firebird V have whiteline following sensors, the grid should be either black lined on white surface or whitelined lined on black surface. Nothing else can be useful here.

   (c) **Mounting the Setup:** As the cleaning setup comprised of one long belt , steel platform , DC motor , Wheel and cleaning brush , So it makes the whole setup very bulky and heavy. Sometimes it leads to misbalancing of bot.

   (d) **Container:** It plays a vital role as small container will require frequent round trips and large container will not , but size of container also imposes design problems as it has to be placed exactly between bot and brush. So we have to use optimum sized container.

   (e) **Position of Brush:** As the bot follow the brush and container will be dragged , So to avoid it we have to lift both brush and container while the bots move.

   (f) **Traversal Direction:** As the bot is following black lines, traversal in diagonal direction not possible. The bot can move only forward,backward,left,right directions. The North, West, South, East should be directives forward grid square positions.

2. **Recharging Bot**

(a) **Metallic Strips for Bot:** Small metallic strips that will serve as charging point for the Bot as they will be connected to the Bot Battery.

(b) **Metallic Strips for Charging Platform:** Metallic strips should be long enough to serve as charging platforms for the Bot as they will come in contact with the metallic strips of the Bot and it will initiate the charging of Bot.

(c) **Position of metallic strips:** Metallic strips should be placed beneath the bot only as it will make the design more generic. If we place metallic anywhere else then it may conflict with the primary goal of bot. (Recharging is add on feature along with existing functionality of bot)

(d) **Traversal Direction:** As the bot is following black lines, traversal in diagonal direction not possible. The bot can move only forward,backward,left,right directions. The North, West, South, East should be directives forward grid square positions.

4 Implementation

We have divided our complexity of design into 2 modules
Module 1: Design of the Cleaning Bot
Module 2: Design of the Charging Bot

4.1 Cleaning Bot

To clean the given path.
Functioning: Bot will extend the arm (brush will be attached) and release the arm as the arm is kept an angle to avoid friction between brush and floor caused during motion. Container will also be released to the floor. Now Arm/brush will be pulled backward to collect any trash into container. Later both the container and arm will be pulled up to avoid friction. Now the bot is free to move.

4.2 Charging Bot

When bot decides that it needs to be recharged then it goes to the Charging platform and charging will start.

Primitive	Use
buzzer_pin_config (void)	Configuration of Buzzer pins
buzzer_on()	Turn the buzzer ON
buzzer_off()	Turn the buzzer OFF
adc_init()	Initialization of ADC hardware
adc_pin_config (void)	ADC hardware pin configuration
ADC_Conversion(unsigned char Ch)	Conversion of unsigned char into Hex value to display on LCD
lcd_port_config (void)	LCD hardware port configuration
print_sensor(row,coloumn,channel)	Function To Print Sesor Values at Desired Row And Coloumn Location on LCD
lcd_cursor(row, column)	Place cursor at given row and column on LCD display
lcd_string(data)	Write null terminated data on LCD display
motion_set (Direction)	Function to set wheels motion with given direction argument
motion_pin_config (void)	Function for configuration of motion pins
ISR(INT4_vect)	Interrupt service routine for INT 4
ISR(INT5_vect)	Interrupt service routine for INT 5
timer5_init()	Timer 5 initialized in PWM mode for velocity control
orient(int value)	Decoding the path variable value for appropriate grid traversal
follow()	Function containing instructions to follow while traversing grid
servo_1(int degrees)	rotates the motor by degrees
servo_1_free (void)	makes servo 1 free rotating
c2_pin_init(void)	to initialize the DC Motor
c2motor_control(char Direction)	to rotate the DC Motor
c2_linear_distance_mm(int DistanceInMM)	C2 specified distance in mm
right_degrees(unsigned int Degrees)	rotate in right direction by degree
left_degrees(unsigned int Degrees)	rotate in left direction by degree
initialize_servo_motors()	initialize the servo motor
sweeper_job()	does the sweeping job

Table 1: Firebird V HAL Primitives

5 Design Diagrams

5.1 FSM and Statecharts

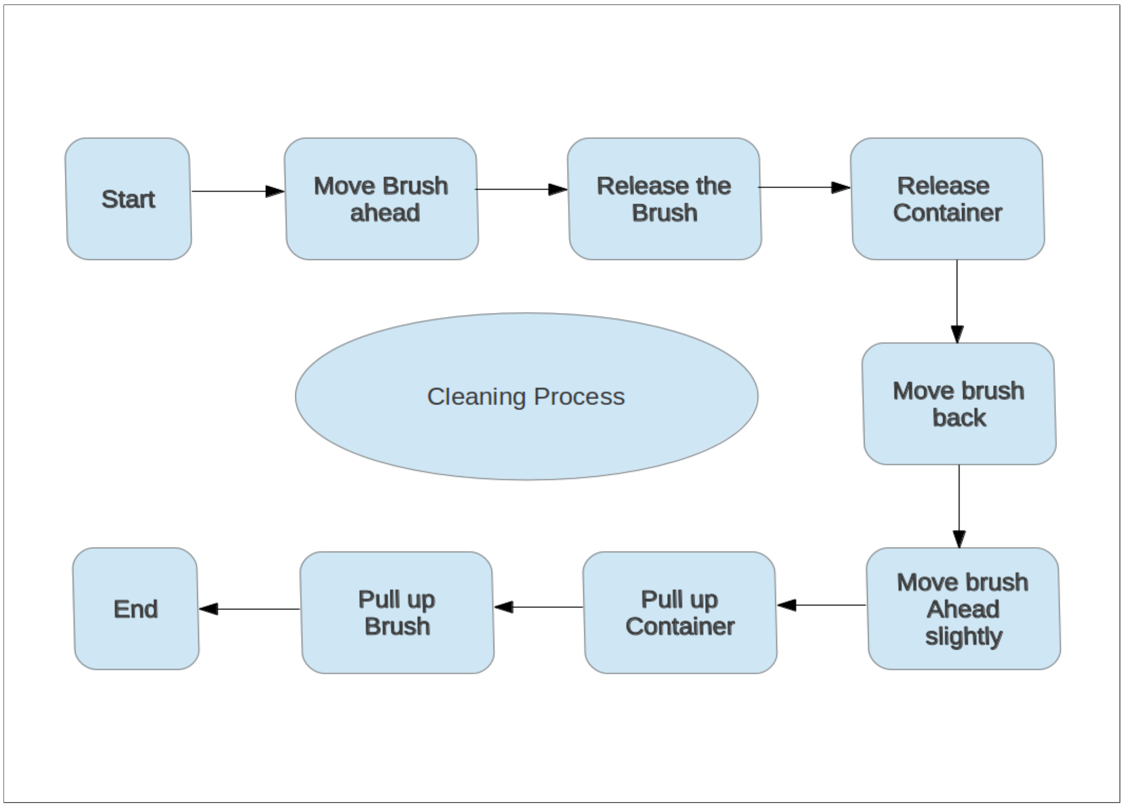

Figure 1: Finite State Machine for Detection & Mapping

6 Testing Strategy and Test Data

Here we tested our design continuously as we used Iterative and prototype based model
We used simple phenomena for testing ”Build and Test Module wise”

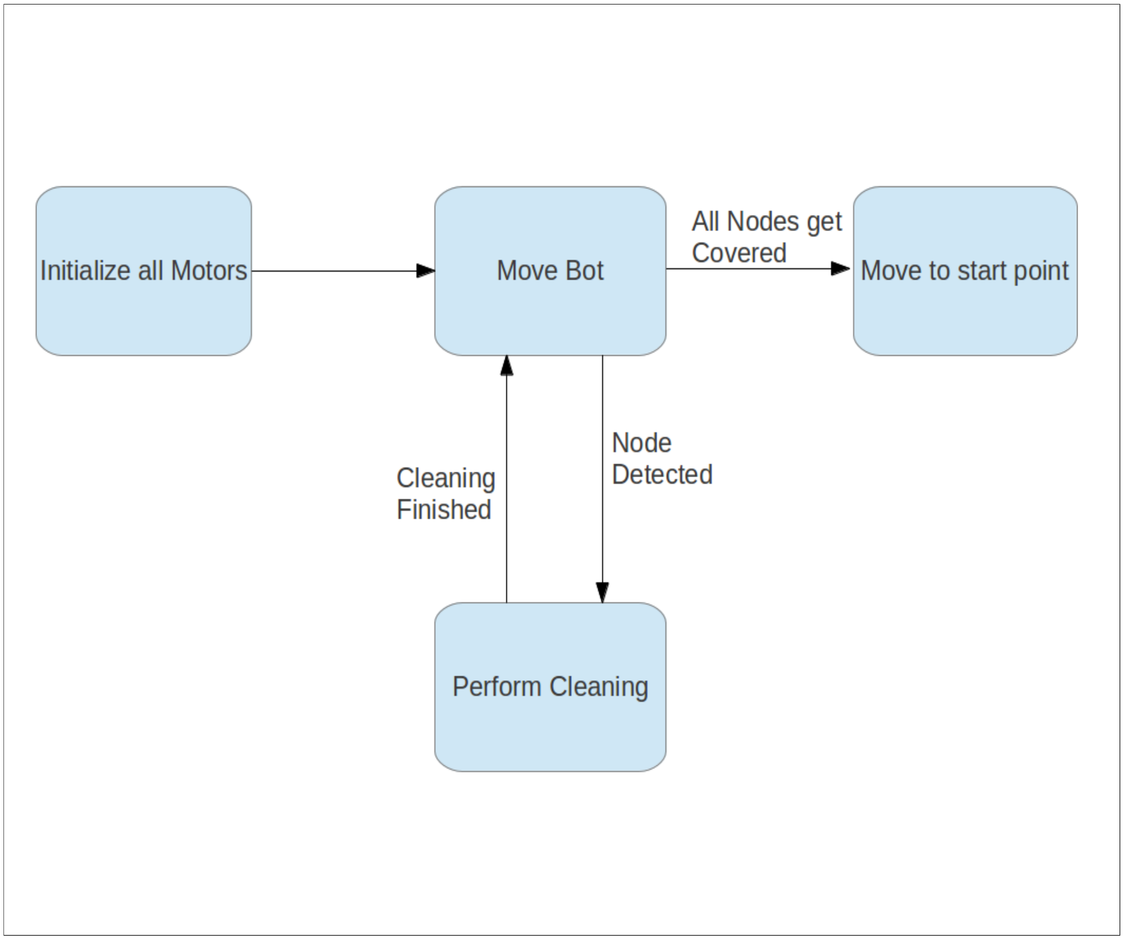

Figure 2: Statechart Diagram

6.1 Testing Cleaning Task

We tested our Bot by running it over given path multiple times .While testing we found that some calibration was required for the motor control and bot movement due to access weight of bot.

6.2 Testing Charging Task

We tested our Bot by forcing the bot to go to charging station (by reducing the battery level below threshold) , the Bot also stored the coordinates of node where it stopped working.

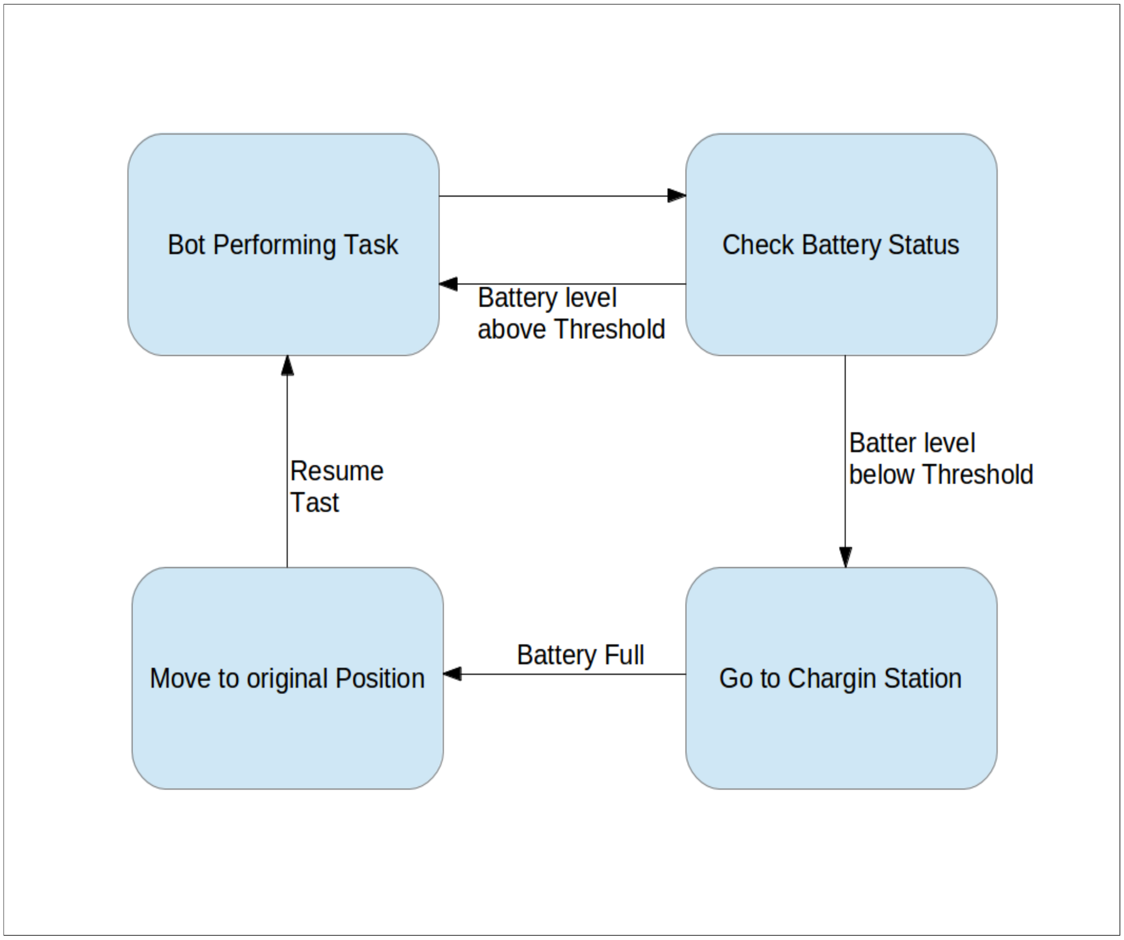

Figure 3: Statechart Diagram

6.3 Testing Grid Follower

We tested our BOT to check whether it's following the black line and covering the entire arena or not . Initially we tested the blackline follower sensors and noted down their readings . We found that the readings of the sensor changes from time to time as the readings depends on the intensity of the light, because of this BOT not following the black line when the intensity of the light is low . We adjusted the forward distance and turning angle so that the BOT will follow the blackline . We also tested for the checkpoints at the cross junction of the blacklines which is helpful in localisation of the BOT

6.4 Verifying the above Tests

We verified the correctness of above tests by manually monitoring them and running tests again and agin several times.

7 Design Challenges

7.1 How to Mount the Setup ?

1. As the **size and weight** of Bot was large so managing the whole setup on top of Bot keeping the balance was a challenge.

7.2 How to make the BOT to follow the grid?

1. **Thickness of the black line??**
   Thickness of the blackline also plays a major role as our BOT has to cover the entire arena depending on the thickness of the line and the cross junctions of the black line whiich are served as check points. One's design has to work even the thickness of the line is thin.

2. **Ability of the sensors to detect the black line**
   One should check the ability of the sensors to distinguish between the black line and white line

3. **Light conditions**(since ability of sensors depends on the Light intensity)
   Readings of the sensors varies depending on the intensity of the light hence one should check the ability of the sensors to distinguish between the black line and white line in different light conditions.The proper sensor calibration has to be done so as to make the system fleible independant of adverse lighting conditions.

7.3 Where to place Container?

1. It plays a vital role as small container will require frequent round trips and large container will not , but size of container also imposes design problems as it has to be placed exactly between bot and brush. So we have to use optimum sized container.

7.4 Where to place Metallic Strips for Bot?

1. Metallic strips should be placed beneath the bot only as it will make the design more generic. If we place metallic anywhere else then it may conflict with the primary goal of bot. (Recharging is add on feature along with existing functionality of bot)

8 Future Work

We have identified following possible improvements for future projects.

- Sheduling the bots lined up to charge their battery at charging platform. eg:- currently we are using single line for charging.
- Implementing an interface to send command to bot for particular action using zigbee wireless communication. eg:- User can dynamically set values for Grid-dimension, add new path , remove or block old path , etc.
- While returning to the charging point bot should be able to compute shortest path excluding paths having any bot.
- We can use one more sensor to monitor the container status and make an instant round-trip.

9 Conclusion

For Cleaning Bot

1. This system demonstrates that it is possible to clean Bot path effectively and efficiently.
2. Size of Trash collected will depend on Bot and its container due to limitation of their size.

For Charging Bot

1. This system demonstrates that Bot can be programmed to monitor its battery status and make a decision when to recharge.

2. In our system Bot can compute shortest path between current position and the Charging Station effectively.

Although real environment of field is rather different from the demonstration platform, this project should still function correctly as a prototype model. In real field, we cannot use grid with black lines because of the soil and the dirt on the floor. However, black lined grid for this project is mere abstraction of any mechanism that allows the Bot to move in field given some guidelines conditions. Here,grid lines serve the purpose of traversal inside the field. We can use GPS or virtual GPS system to move along straight line in real environment. This change merely needs modification at HAL layer code [4.2]. Rest of the system remains the same.

Real time Embedded systems can possess hardware inaccuracies and physical measurement faults.Still, upto certain constraint, the working system can be built. Module based development approach can lead towards complex real-time systems.

References

[1] E-yantra website http://www.e-yantra.org

[2] FireBird V Atmega2560 Robotic Research Platform Hardware manual.IIT Bombay and NEX Robotics Pvt. Ltd.

[3] FireBird V Atmega2560 Robotic Research Platform Software manual.IIT Bombay and NEX Robotics Pvt. Ltd.

[4] GNUPlot tool http://www.gnuplot.info