CS684 Course Project

Movement Control and Obstacle Avoidance of Hexapod

Group-02
Meruva Naga Sreenivas-10305079
Krishna Teja V-10305029
Nikhil M-10305032

Table of Contents

1 Introduction

In our project, we control the hexapod motion. Initially the hexapod moves in straight line. Hexapod is fixed with a proximity sensor in front, right and left whenever it encounters an obstacle in front, it checks the right and left respectively and turns accordingly. If there is obstacle in all directions , it turns around and moves. We wrote an Esterel interface, which provides various signals and helps for the motion of the hexapod.

Product Perspective: This project which helps in controlling the motion of hexapod is really useful in applications which requires bots in uneven surfaces, hilly areas etc.

Product Functions: The function is basically movement control of the hexapod. The final product makes the hexapod to move in different directions avoiding obstacles. The esterel interface written will be helpful in developing further applications on hexapod using esterel.

User Characteristics: While in the obstacle avoidance, user expects the bot to detect the obstacles and move accordingly to avoid them.

Constraints: Accuracy of the proximity sensor to detect obstacles. Power consumption of hexapod. Accuracy of hexapod motion in particular direction.

Assumptions and Dependencies: We assume that the readings are appropriate which depends on accuracy of proximity sensor. We assume even size of legs. We also assume static obstacles.

Details

1.1 Supportability

Since we are writing an esterel interface for the motion of the hexapod. This interface can be used in any application of hexapod. This esteral interface can be used directly in developing other projects on hexapods.

1.2 On-line User Documentation and Help System Requirements

We use documentation which clearly describes the architecture of hexapod and also some documentation which provides thorough description of esterel.

2 Problem Statement

The main goal of the project is to make hexapod move in such a way that it detects the obstacles and changes its motion in such a way that it avoids those obstacles

3 Requirements

Hardware requirements:

1. Hexapod

2. Proximity sensors

Software Requirements

1. Esterel Compiler

2. Windows, Linux- Wndows to compile c code generated and burn hex file. Linux to compile esterel code and generate c code for esterel code written

3. AVR studio to burn hex file on hexapod

4. ICC AVR for compiling c code to generate hex file

Functional Requirements:

1. Hexapod should be able to move in different directions. It should be able to move in a specific direction accurately.

2. Proximity sensors should be able to sense the presence of obstacle as precise as possible and then bot should move accordingly to avoid the obstacle.

3. Hexapod should move with a specified speed and direction.

Non Functional Requirements:

1. Minimize power consumption

2. Increase accuracy of the motion of hexapod (Hexapod should move as straight as possible)

3. Decrease error readings due to interruption of legs to proximity sensor

4 Implementation

Algorithm Implemented:

1. Initially hexapod moves in a forward direction, it senses the presence of any obstacle using the Front sharp sensor.
2. If it does not find any obstacle the hexapod continues to move in forward direction
3. If it finds an obstacle, it checks the presence of any obstacle in the right side using the Right Sharp sensor.
4. If there is no obstacle in the right hand side of the hexapod. It turns towards right and continues to move forward. If it finds a obstacle it checks the presence of any obstacle in the left side using the Leftt Sharp sensor.
5. If there is no obstacle then it turns towards left and continues to move forward
6. If there is an obstacle in the left side also then the hexapod turns around by 180 degrees (turns back) and then continues the same procedure

Esterel Interfacing

We have designed an esterel interface for this project. This esterel interface for the hexapod helps in writing the code in esterel and make this run on the hexapod. This esterel interface provides various signals for performing various functions by hexapod.

A code written in esterel is compiled using an esterel compiler along with the interface we provided. This generates a C code. This C code is then compiled to generate a hex file. The hex file generated is then burnt on the hexapod.

There are a variety of signals provided by the interface. They are listed as follows:

INPUT SIGNALS

1. FRONT_IR_VALUE(integer) : Gives the value read by the front proximity sensor
2. FRONT_LEFT_IR_VALUE(integer): Gives the value read by the proximity sensor present in between front and left proximity sensor
3. FRONT_RIGHT_IR_VALUE(integer): Gives the value read by the proximity sensor present in between front and right proximity sensor
4. REAR_IR_VALUE(integer) : Gives the value read by the rear proximity sensor
5. REAR_LEFT_IR_VALUE(integer): Gives the value read by the proximity sensor present in between rear and left proximity sensor
6. REAR_RIGHT_IR_VALUE(integer): Gives the value read by the proximity sensor present in between rear and right proximity sensor
7. LEFT_IR_VALUE(integer): Gives the value read by the left proximity sensor
8. RIGHT_IR_VALUE(integer): Gives the value read by the right proximity sensor
9. FRONT_SHARP_VALUE(integer) : Gives the value read by the front sharp sensor
10. FRONT_LEFT_SHARP_VALUE(integer): Gives the value read by the sharp sensor present in between front and left sharp sensors
11. FRONT_RIGHT_SHARP_VALUE(integer): Gives the value read by the sharp sensor present in between front and right sharp sensors
12. LEFT_SHARP_VALUE(integer): Gives the value read by the left sharp sensor
13. RIGHT_SHARP_VALUE(integer): Gives the value read by the right sharp sensor

OUTPUT SIGNALS

14. BUZZER_ON: Instructs hexapod to give a buzzer
15. BUZZER_OFF: Instructs hexapod to off the buzzer sound
16. MOTOR_[ij](integer): This instructs the ith leg jth motor of hexapod to move by the angle specified as integer. So I ranges from 1 to 6 and j can be A,B,C. For eg, MOTOR_1A(45) instructs hexapod to move motor 1A by 45 degrees.
17. MOVE_FWD: Instructs the hexapod to move forward (2 leg motion)
18. MOVE_REV: Instructs the hexapod to move backward(2 leg motion)
19. ROTATE_RIGHT(integer): Instructs hexapod to rotate rightwards specified number of times. For eg, ROTATE_RIGHT(8) instructs hexapod to rotate right 8 times
20. ROTATE_LEFT(integer): Instructs hexapod to rotate leftwards specified number of times. For eg, ROTATE_LEFT(8) instructs hexapod to rotate left 8 times
21. LCD_INIT: Initializes the LCD display
22. LCD_SET_4_BIT: Sets LCD display to 4 bit mode
23. LCD_CLEAR: Clears LCD screen
24. LCD_DISPLAY_1(string): Displays a string in the first line of LCD display starting from 1,1
25. LCD_DISPLAY_2(string): Displays a string in the second line of LCD display starting from 2,1
26. LCD_DISPLAY_INT_1(string): Displays an integer in the first line of LCD display starting from 1,1
27. LCD_DISPLAY_INT_2(string): Displays an integer in the second line of LCD display starting from 2,1
28. SERVO_CALIBARATION: Sets all motors to 90 degrees mainly useful in motor level abstraction

29. DELAY(integer): Gives delay specified in milli seconds. Prefered delay values are 100, 250 500 1000. For other values it t takes a delay of 100ms. Mainly useful in motor level abstraction

Levels of abstraction

1. Total hexapod level: Using signals like MOVE_FWD etc. This abstraction is mainly helpful in applications where developer can concentrate on other aspects of the application rather than on motion. Applications like capturing images and sending them to remote area. In this programmer need not concentrate much on motion of hexapod legs, so he can use this abstraction, rather than he can concentrate on data communication through zigbee etc.

2. Motor level abstraction: This abstraction helps in projects where people want to develop applications like hexapod climbing on stairs, dancing etc. where there is a heavy need of motion of each motor.

5 Testing Strategy and Data

We tested the bot with different sized obstacles and then checked how the bot is able to detect them. However the obstacle should be at least as high as leg of the hexapod to be detected by proximity sensor.

We also test the motion with a number of obstacles placed at different places in the motion of the bot and check how hexapod is avoiding them. We also check the motion of hexapod when multiple objects or an object with large width is placed as obstacle.

We created a arena in a way such that every motion of hexapod like forward motion,turning left,turning right and turning back can be seen there.

6 Challenges faced

1. Power consumption
2. Less height obstacles: Minimum height should be that of the hexapod
3. Sharp obstacles present in between 2 sensors (front or left and front or right) . These goes undetected
4. Uneven surfaces and uneven legs of bot
5. Motion will be unpredictable when obstacles are dynamically changing.

7 Future Scope

1. More applications can be built using this esterel interface

2. Directly use this project in applications which require motion of hexapod avoiding obstacles and do some additional work

8 References

1. Esterel Documentation- Esterel v5.21 System Manual- G.Berry and Esterel team
2. Hexapod Documentation Firebird-V- Hexapod hardware manual
3. Esterel interfacing for Firebird-V-atmega2560