Journal of Computer and Communications, 2014, 2, 8-16
Published Online May 2014 in SciRes. http://www.scirp.org/journal/jcc
http://dx.doi.org/10.4236/jcc.2014.27002

Design and Implementation of a Multi-Sensor Based Object Detecting and Removing Autonomous Robot Exploration System

Fan Wu[1], Johnathan Williams[2]

[1]Computer Science Department, Tuskegee University, Tuskegee, AL, USA
[2]Electrical Engineering Department, Tuskegee University, Tuskegee, AL, USA
Email: wuf@mytu.tuskegee.edu, jwilliams6312@mytu.tuskegee.edu

Received March 2014

Abstract

Developing autonomous mobile robot system has been a hot topic in AI area. With recent advances in technology, autonomous robots are attracting more and more attention worldwide, and there are a lot of ongoing research and development activities in both industry and academia. In complex ground environment, obstacles positions are uncertain. Path finding for robots in such environment is very hot issues currently. In this paper, we present the design and implementation of a multi-sensor based object detecting and moving autonomous robot exploration system, 4RE, with the VEX robotics design system. With the goals of object detecting and removing in complex ground environment with different obstacles, a novel object detecting and removing algorithms is proposed and implemented. Experimental results indicate that our robot system with our object detecting and removing algorithm can effectively detect the obstacles on the path and remove them in complex ground environment and avoid collision with the obstacles.

Keywords

Autonomous Robot Exploration System, Object Detecting and Removing Algorithm, Multiple Sensors

1. Introduction

Developing autonomous mobile robot system has been a hot topic in AI area [1]. Many ideas have been proposed and applied to autonomous robots. For example, Stanford Research Institute International AI Center developed Saphira [2]; Carnegie Mellon University Robot Institute developed Teambots [3]. Today autonomous robots are attracting more and more attention worldwide in both academia and industry. The study of robotics has increased considerably in the last decades and thanks to ongoing research both in the industry and education, robots have become commercially available to the general public. Contests of football robotic teams [4], and similar, are held annually and autonomous robots, as pets and home devices, such as dogs and vacuum cleaners, are already available commercially. Some companies like Lego or VEX Robotics have commercial robotics kits that contain everything you need to create your own robotic system.

Designing a robot's behavior is a challenging task that researchers have been working for a long time. The behavior-based approach to robot control has been the basis for many implemented robotics systems [5]-[7]. A simple task such as moving an object from one point to another implies the intervention of a set of abilities that a robot must be provided with in order to successfully achieve its goals. Some of these are: ability to perceive its environment, ability to make decisions when planning the task, ability to navigate through the environment and to avoid obstacles during the navigation, ability to execute the planned actions, and ability to recover from failure, among others. Clearly, the complexity of each individual's ability, and therefore the overall robot's behavior design, is related to the complexity of the environment where the robot carries out the task. The higher the complexity of the environment, the more challenging results the robot's behavior design. Besides, the robot features also play an important role in the robot's performance of the task.

Path finding is an important issue and one of the most fundamental problems in mobile robotics [8] [9]. It is to find a most reasonable collision-free path for a mobile robot to move from a start location to a destination in an environment with obstacles. This path is generally optimal in some aspects, such as distance or time. However, different regions usually have different road conditions, such as sandlot, grassplot and so on. How to find a reasonable path meeting the need of above criterions and escaping from obstacles in such complex ground environment is an open problem.

In this paper, we describe our experience on designing and implementing a multi-sensor based object detecting and removing autonomous robot exploration system. The robot system described in this paper, 4RE, was designed and implemented with the VEX Robotics Design System [10]. The VEX Robotics Design System is divided into seven different "subsystems" where each one of the subsystems plays an important role in the design. Once designed, all the systems were implemented together to create 4RE exploration robotic system. The programming software used to program the robot was easyC. EasyC is a drag and drop application with predefined functions in the C language.

The rest of the paper is organized as follow: Section 2 describes the VEX robotics design system briefly; Section 3 presents our hardware architecture and the system design process; Algorithm design is presented in Section 4 and our experimental results are discussed in Section 5. Finally section 6 concludes this paper with our future directions.

2. VEX Robotics Design System

The VEX Robotics Design System is divided in to seven different subsystems that interact with each other to create a robotic system. In order to design and implement a robot, some knowledge about each subsystem is required, which are presented in the following subsections [11].

2.1. Structure

The structure subsystem is very important because it forms the base structure of the robot. It can be considered to be the skeleton of the robot, where the other subsystems are attached.

The structural components that form this subsystem include metal sheets, frames, nuts, screws, standoffs, and more building parts. The VEX Robotics Design System comes with pieces of different shapes and sizes. However, it follows a standard hole-spacing system that expands the user's options to create whatever shape is needed.

The design of this subsystem has to be strong and firm but the weight of the robot always has to be kept in mind depending on the robot's tasks and challenges. Another important thing to consider is that the other subsystems will be added to the structure; therefore space has to be measured to successfully implement the remaining parts.

2.2. Motion

The motion subsystem is in charge of making the robot move, in other words it is like the muscles in our body. Motion includes the motors that generate power as well as the wheels and gears that move the robot around, transforming the power into work to help the robot achieve its goals. Just like with the Structure Subsystem, wheels, gears, motors, and servomotors come with standardized holes that make the assembling process easier and faster. This property is called openness and it gives the user the ability to add hardware without having to make changes, or as less change as possible.

The design of this subsystem will vary depending on the needs of the robot and it will be strongly incorporated with the components of the Structure Subsystem. The number of wheels used on the base to move the robot and the number of motors to power the wheels will depend on how heavy or how big the structure of the robot is. Depending on the robot's tasks; motors and gears need to be considered to create arms or other moving parts on the robot's structure.

2.3. Sensors

The Sensor Subsystem is what helps the robot detect different things in its environment. It is the eyes and ears of the robot. Without this subsystem it would be impossible for a robot to operate autonomously. The VEX Robotics Design System includes some sensors in the kit but different types are available online. The sensors included are bumper sensors, limit-switch sensors, light sensors, ultrasonic sensors, and infrared light sensors.

The design of this subsystem starts with the structure because there are some sensors that require a specific placement. Once the sensors have been installed, they will draw the power indirectly from the batteries through the Microcontroller. Finally, The Microcontroller will receive the information from the sensors making the robot behave depending on the code being executed.

2.4. Logic

The Logic Subsystem is the major component of the VEX Robotics Design System. The Microcontroller is the only part that forms this subsystem but it is in charge of coordinating and controlling all the other subsystems. In other words, it is the brain of the robot.

The Microcontroller interacts directly with the other subsystems. When designing the structure, a safe place for the Microcontroller has to be considered because most of the other subsystems need to be connected to it. The power from the batteries is drawn directly from the Microcontroller and it is then distributed among the other subsystems. Once powered, the Microcontroller will control all the motors directly and process the information gathered by the sensors.

2.5. Programming

The Programming Subsystem is not a physical system of VEX. It is the software application that tells the robot how to behave. The programming software included in the VEX Robotics Design System is easyC. EasyC is a drag-and-drop application with predefined functions in the C language, as shown in **Figure 1**. The functions are "dropped" in a programming window and depending on the function, a window prompts asking the user to enter the desired arguments. EasyC is a very easy to use tool but it seems to be designed for users with basic programming skills.

3. Hardware Architecture

3.1. System Architecture

Define abbreviations and acronyms the first time they are the VEX Robotics vehicle is composed of several different components that interact with one another in recognizing an object in front of the vehicle, picking it up, and setting it back down again. All of these subsystem components of the robot consist of the wheel and motor system that moves the machine, the gear system that controls the movement of the arm, the robot arm itself, and the attached claw device the can extend, grab, pick up, and set down objects. Each individual subsystem can be further broken down into smaller details further in this report. In **Figure 2**, a visual representation of the robot is shown. VEX Robotic tools and machinery, including metal frames, gears, nuts, screws, plastic wheels, motors,

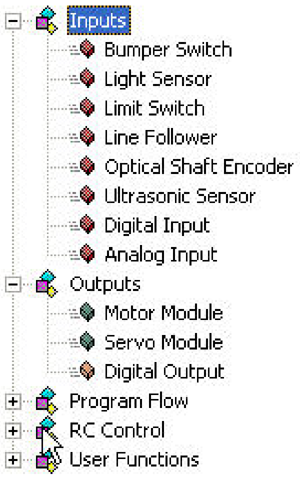

Figure 1. Some predefined functions in EasyC.

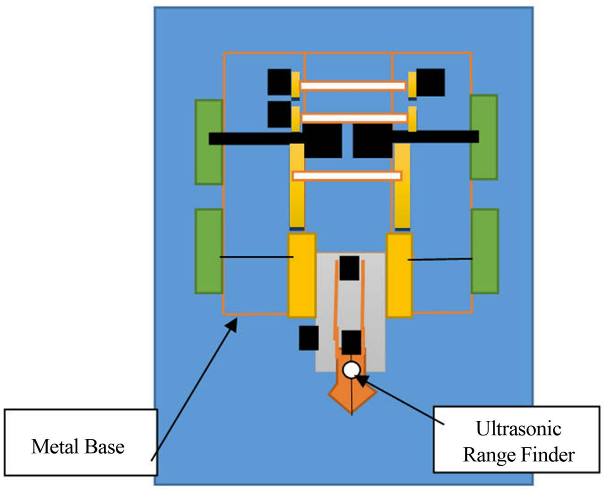

Figure 2. Hardware Architecture of VEX Robot. 1. Black—Motors; 2. Green—4 inch Wheels; 3. Yellow—Gear System; 4. White—Gear Rods/Axes; 5. Grey—Robot Arm; 6. Orange—Robot Claw with Elbow Joint in Rest Position.

and the Cortex microcontroller. The claw itself came in a separate kit already assembled with its own motor and simply needed to be attached to the arm.

The machine itself utilizes four large 4 inch plastic wheels that provide a strong base and balance for the robot during movement. Eight motors are used in total with two of them being used to control the wheels.

Two pairs of the smaller gears are used to drive the four large gears on the robot arm with the last two gears controlling the up and down movement of the robot arm.

For the sensors, three are used in total with two of them being the limit switches to control the range of motion of the arm and the third being an ultrasonic range finder that is attached on the claw component.

All of these various parts are wired to a single VEX Robotic Cortex microcontroller that acts as the "brain" of the robot where it processes the code downloaded into its memory and follows the programming instructions.

3.2. System Design and Implementations

The robot is composed of various simple subsystems that when programmed as one, is able to accomplish different tasks, in this case being the simple objective of lifting and setting down an object in its path.

1) Base

Obviously one of the most important aspects of designing and building an object is constructing its frame or base in which it will support the entire weight of the machine itself. A good strong base is necessary in robotics as it provides structural integrity and durability when it's out in the field accomplishing its assigned objectives. The material used to create the base was several near foot long metal frames or beams that form a square shape. The parts are screwed tightly together, ensuring that it can handle the stress and load of the other VEX parts that would later be built on top along.

2) Wheels and Motors

The next stage of the building process was the appropriate selection and positioning of the wheels and motors. As mentioned before four 4 inch plastic wheels were implemented in this design as they provided the robot with enough structural support and weight when it is traveling over various surfaces. Other wheel types were attempted before this and failed poorly as they simply were not big enough to hold the robot's weight required, resulting in the vehicle tipping over whenever it attempted to move. Also instead of four motors being used on the wheels individually, two were decided to be adequate as they would rotate the back wheels, and providing the necessary amount of force required to move the entire machine while the front two wheels give additional stability for the robot's frame and as well as aiding its movements whether forwards or backwards as well.

3) Gear System

A four pair gear system is necessary in controlling the up and down motion of the robot arm. Two pairs of smaller gears are "driver" gears as they provide the initial torque or turning force, courtesy of the motors attached to them, to rotate the two pairs of larger gears that are connected to the arm component. At first only a two and later three pair gear system was used in this design, however it was quickly shown that the given torque would not be sufficient in raising and lowering the robot as the total weight of the arm component and the potential load it might carry was simply too heavy. However when a fourth pair of smaller type gears were provided, the gear system was successfully able to lift the arm up and down smoothly. Also, in conjunction with the gear system, three motors were used to provide the torque, with a primary motor and two backups, a secondary and tertiary motor. The primary and secondary motors were attached to the last row of gears as they could give the system the most turning force when working together. The third motor was connected to the third row of gears as any extra torque it gave would simply be insurance in case the load the arm was carrying was on the heavy side.

4) Arm

Attached to the gear system was the robot arm itself. This section of the robot might have been the most complicated and difficult area of the robot to design, build, and program as it consisted of the three smaller subsystems that worked as one to provide the robot with a range of motion and abilities.

a) Arm bar

The "arm bar" of the robot was the heaviest part of the entire arm subsystem. It was directly attached to the last pair of the gear system as it would be needed to raise or lower the whole arm component itself. In essence one can liken it to the upper arm that is connected to the shoulder joint of the human body.

b) Elbow Joint

The second subsystem of the arm would be the "elbow joint" as similar to a human body it would bend like an elbow to raise or lower the forearm. When the arm bar is raised, the elbow joint can adjust the number of degrees it bends downwards to aim the VEX robotic claw. In its rest position it lies flush on top of the arm bar.

c) Arm Extender/Claw

The final part of the robot arm is the arm extender and claw device that is attached to the elbow joint of the robot. After the arm bar has either risen or lowered and the elbow joint has moved the appropriate amount of degrees, the arm extender simply stretches outwards from the elbow joint with the claw attached at its front already open and ready to grab the object in the robot's way. Once the item is in ensnared, the arm can simply be pulled back into the elbow joint, where it then soon returns to its rest position. This process can easily draw comparisons with the claw game machines that one utilizes to grab prizes in a box.

5) Sensors

For this project only three sensors were needed. There were two limit switches and an ultrasonic range finder. The limit switches, were sensors that made a slight clicking nose when their bar is pressed at the right angle. These are extremely useful in telling the robot the arm when to stop moving. One limit switch was attached on a base bar just under the arm bar part to halt its movement with the second connected below the elbow joint when it was in its rest position. The ultrasonic sensor contains two speakers in which one sends out a frequency pulse and the second retrieves it. There is also a timer installed as it notes the time required for the sound travel back to the sensor, and later takes that information and calculates the distance the frequency traveled. With this attached to the claw, the robot was able to easily detect whenever an item stood in its way.

6) Microcontroller and Battery

The VEX Cortex Microcontroller is the device that makes all of the robots actions possible. As mentioned before it "computes" or processes the commands it downloads, via USB cable, from the Easy CV4 language. It along with the 7.2 V battery is attached to an empty space in the back of the robot. The motors are wired in motor ports 1 - 10 while the sensors are wired in digital input/output ports 1 - 12.

4. Obstacle Detecting and Removing Algorithm

The programming itself was somewhat simple and not really complicated although the lines of code was extensive. The code basically instructed the robot to begin traveling in a straight line. However as soon as the ultrasonic range finder detects an object in its direction, the robot will immediately stop, back up slightly. The next stage is critical as the robot will soon start to grab the object as it uses all three motors to raise the arm bar itself to an appropriate angle. Next the program has the robot bend its elbow joint downwards where it will then open the law to its maximum width and extend the arm downward where it will stop and the claw will close shut with the object in its grasp. The elbow joint will then rise upwards until it hits the limit switch, signifying its "rest" position. The arm extension will then immediately return to its shortened position with the object still in the hold of the claw. At this stage there are several options one can make in the programming for the robot. The programmer can either change the last bit of code where it has the machine lower the arm and set the object back down and move away or simply spin in other direction and then set the object down in a different spot. Either one is acceptable as the robot demonstrates the ability to identify an obstacle, grab, and lift it. The code that will be shown later in this report will be the former, where it simply sets the object back down and moves away.

Figure 3 shows the data flow of our object detecting and removing algorithm.

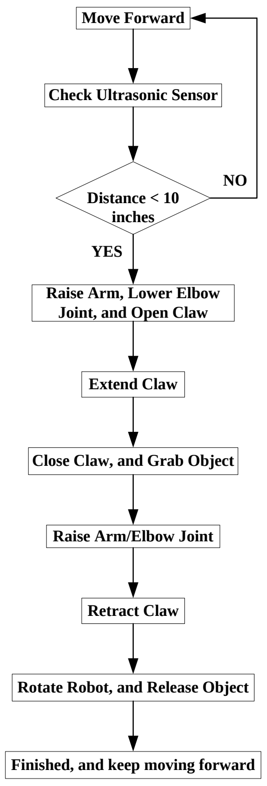
Move Forward
Check Ultrasonic Sensor
Distance < 10 inches
NO
YES
Raise Arm, Lower Elbow Joint, and Open Claw
Extend Claw
Close Claw, and Grab Object
Raise Arm/Elbow Joint
Retract Claw
Rotate Robot, and Release Object
Finished, and keep moving forward

Figure 3. The flowchart of the Algorithm.

The algorithm 1 labeled “Grab Obstacle Code” shows the pseudo code of our obstacle detecting and removing algorithm.

Alg. 1: Grab Obstacle Code

```
1.  SetMotor (3, −30)
2.  SetMotor (6, 35)
3.  while (1) //condition is always true
4.  while (1)
5.  Start Ultrasonic (1, 11)
6.  sound = GetUltrasonic (1, 11)
7.  if(sound > 10) //object is far away
8.  printToScreen(“go\n”)
9.  SetMotor (3, −30)
10. SetMotor (6, 35)
11. else (
12. SetMotor (3, 0)
13. SetMotor (6, 0)
14. break //jump to next while loop
15. while (1)
16. printToScreen (“pickup\up”)
17. SetMotor (3, 0)
18. SetMotor (6,0)
19. SetMotor (3, 40)
20. SetMotor (6, −45)
21. wait (400) //wait for .4 seconds
22. SetMotor (3, 0)
23. SetMotor (6, 0)
24. SetMotor (1, −127) //raise arm
25. Wait (500)
26. SetMotor (9, 80)    //raise arm
27. Wait (100)
28. Break
29. While(1)
30. SetMotor (7, 70) //lower elbow joint
31. Wait (550)
32. SetMotor (7, 0)
33. SetMotor (2, 0)
34. SetMotor (9, 0)
35. SetMotor (4, −60) //open claw
36. Wait (1000)
37. SetMotor (4, 0)
38. SetMotor (8, −80) //extend arm
39. Wait (750)
40. SetMotor (8, 0)
41. SetMotor (4, 60) //close claw
42. Wait (1000)
43. SetMotor (4,0)
44. SetMotor (8, 85) //shorten arm
45. Wait (1200)
46. SetMotor (8, 0)
47. Break
48. While(1)
49. Click = GetDigitalInput (3)
50. If (click == 1)
```

```
SetMotor (7, −120) //go to rest position
Else (
Break)
While(1)
SetMotor (3, 45) //rotate robot
SetMotor (6, 40) //rotate robot
Wait (500)
Break
While (1)
Hit = GetDigitalInput (2)
If (hit == 1)
SetMotor (1, 100) //lower arm
Else (
SetMotor (4, −60) //open claw
Wait (1000)
SetMotor (4, 0)
SetMotor (3, 40) //reverse
SetMotor (6, −45) //reverse
Wait (400)
SetMotor (4, 60) //close claw
Wait (400)
Break
While (1)
SetMotor (3, −45) //rotate robot
SetMotor (6, −40) //rotate robot
Wait (1000)
SetMotor (3, 0)
SetMotor (6, 0)
Break
While (1)
SetMotor (7, 0) //end program
SetMotor (2, 0) //end program
SetMotor (9, 0) //end program
```

5. Experimental Results

We constructed an environment to test our algorithm. The first environment consists of an open space surrounded by four walls and a line on the ground, as **Figure 4** shows. This environment is designed to test our obstacle detecting and removing algorithm we designed above. Many trials and tests were ran to figure out the appropriate rotating speeds, measured distances to objects, and correct arm movement to grab and pick up the desired object. The robot is best suited to accomplish this task on an even and flat surface where it can easily detect and get to an object. As of right now the robot will only sense objects that are in the path of the ultrasonic sensor. As of now it does not sense obstacles on its flanks. Also the code is easily adjustable when one is trying to make the robot grab objects of various heights, and lengths. The experimental results show that our robot can detect the obstacles on the path precisely and remove the obstacles automatically with our algorithm.

The 4RE robot system is shown in **Figure 5**.

6. Conclusion and Future Work

In this paper we have designed, created, and programmed a robot using a wide array of sensors and motors to detect foreign objects in its path of direction and move the offending item out of the way. It was a struggle first, but the robot was successful in grabbing and setting the object out of its way. It should have no problem accomplishing this task with several types of different items provided they are light of weight and not to large or small.

However, this may present future problems in a real world situation. Since not all will be so easily removed

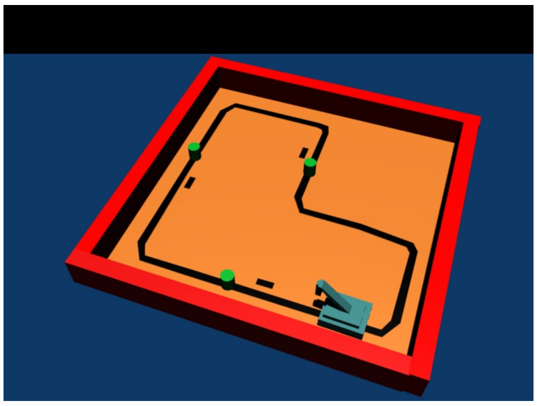

Figure 4. Test Environment for the algorithm.

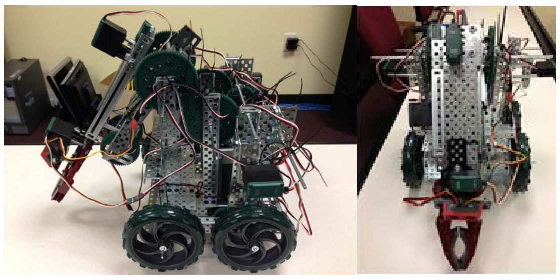

Figure 5. The 4RE robot system.

from their positions as they may come in various shapes, weights, and sizes that would make it difficult for the robot as its sensors would need to be adjusted to distinguish various objects and the best way to move them.

Future work will involve adding more types of resistors to overcome these limits along with improvements to the current robot's design so that any perceived flaws or hindrances to its ability will be immediately corrected.

References

[1] Fernandez, J.A. and Gonzalez, J. (1999) The NEXUS Open System for Integrating Robotics Software. Robotics and Computer-Integrated Manufacturing, 15, 431-440. http://dx.doi.org/10.1016/S0736-5845(99)00037-X

[2] Konolige, K. and Myers, K. (1998) The Saphira Architecture for Autonomous Mobile Robots, SRI International.

[3] T. Balch, TeamBots, 2000. www.teambots.org

[4] Lenser, S., Bruce, J. and Veloso, M. (2001) CMPack: A Complete Software System for Autonomous Legged Soccer Robots. Proceedings of AGENTS'01, ACM Press, 204-211.

[5] Balch, T. andArkin, R.C. (2002) Behavior-Based Formation Control for Multi-Robot Teams. *IEEE Transactions on Robotics and Automation*, **14**, 926-939.

[6] Mataric, M. (1997) Behavior-Based Control: Example from Navigation, Learning, and Group Behaviors. *Journal of Experimental and Theoretical AI*, **9**, 323-336. http://dx.doi.org/10.1080/095281397147149

[7] de Leeuw, J.R. and Livingston, K.R. (2009) A Self-Organizing Autonomous Prediction System for Controlling Mobile Robots. *International Conference on Automation, Robotics and Control System*, 123-129.

[8] Cox, I.J. and Wilfong, G.T. (1990) Autonomous Robot Vehicles, Springer Verlag.

[9] Borenstein, J., Everett, B. and Feng, L. (1996) Navigating Mobile Robots: Systems and Techniques, A. K. Peters, Ltd., Wellesley.

[10] VEX Robotics Design Systems. http://www.vexrobotics.com.

[11] VEX Inventor's Guide. http://www.vexforum.com/wiki/index.php/Inventor%27s_Guide