Embedded Control Systems Project - Collision Avoidance

Prabhu Mani and Jayashankar Manohar,
Master programme in embedded systems, Uppsala University,
Prabhum234@gmail.com,Jayashankar27@gmail.com
Project duration: 31.10.2011 to 23.01.2012

Table of Contents

1. Abstract

Several independent autonomous mobile robots explore bounded occluded scenery. Design and implement three different control algorithms preventing the robots from colliding with stationary and mobile objects.

2. Introduction

The problem is to build autonomous robots and explore a bounded area. There is no fixed target or path for the robot. The destination and path is completely random but the objective is to avoid any static or moving objects. Though the path is random, controllers are implemented to make sure the robots behave as per the algorithm when encountered with any obstacle. To detect obstacle sonar sensors are used. It produces the sound waves and receives the waves reflected back by the obstacle. It calculates the distance of the obstacle by calculating the time difference between generated waves to reflect back to the source. However this sensing solution is not robust.

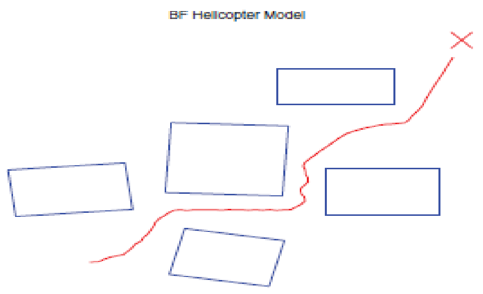
Three different approaches are implemented for avoiding the obstacle. Different obstacle sensing solutions are analysed and implemented on top of the collision avoidance algorithms. Basic reflexive collision avoidance systems are implemented, in which the robot avoids the obstacle when it is in the avoidance range.

3. State of the art

The state of the art in collision avoidance for two different types of algorithms is described in works of Matt Garrat, Michael Hoy and Andrey Savki [1].

a. Boundary following algorithm

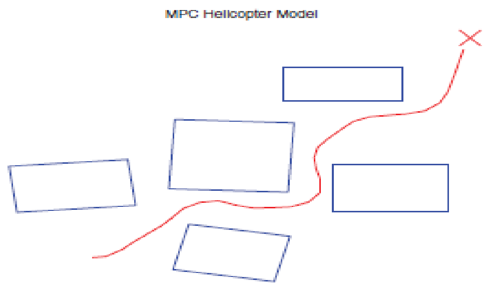
The boundary following algorithm modifies the simple sliding mode control law. The simple sliding mode controller which uses only the minimum distance to the obstacle as input to control law. In boundary following algorithm, when the obstacle is absent, the controller drives the vehicle towards the target point. When the obstacle is encountered, it maintains the minimum distance from obstacle as mentioned in simple sliding mode and the robot tend to follow the obstacle boundary until the vehicle reaches the target heading.



Results obtained from the boundary following algorithm [1].

b. Model predictive algorithm

In model predictive algorithm, the future trajectory is calculated as the sequence of way points to ensure robust control. This algorithm calculates multiple way points. The trajectory is checked for interference against the obstacle. If interference is predicted then two alternative trajectories are generated – one going towards the left of the target and one going towards the right of the target. If there is no viable trajectory due to environment then the trajectories are recalculated. If there is no further path then the vehicle back tracks.



Results obtained from trajectory using the full path planning controller [1].

4. Proposed Approaches

a. Approach1

Design and implement reflexive collision avoidance algorithm. Two sonar sensors scan for obstacle around it. If any obstacle is encountered, turn the robot to desired angle such that robot aligns itself parallel to the obstacle. Simple PID controller is used for turning the car to desired angle and another PID controller for rotating sonar sensor back and forth in a specified range of angles.

b. Approach 2

Design and implement boundary following algorithm, where two sonar sensors, fixed on the edges of the car, scan for obstacle around it. By knowing the surrounding environment, turn the robot gradually until the obstacle is disappeared. A simple PID is implemented for turning the car and another PID controller for rotating sonar sensors back and forth in a specified range of angles.

c. Approach 3

Design and implement collision avoidance algorithm using a new sensor solution, combination of camera and laser beams. The camera sensor is capable of tracking objects of eight different colours which can be pre configured. Laser source is mounted on the vehicle, pointing in front of the cars path of motion. If the robot encounters any obstacle in front of it, the laser point on the path will be disturbed by the obstacle or in other words the laser point will be blocked by the obstacle from projecting on the path of the car. This dislocation of the laser pointer from the camera view could be detected by the camera sensor mounted on the robot. The distance with which an obstacle should be avoided can be configured by adjusting the laser projection nearer or farther to the car.

5. Tools Used for the development

a. Hardwares

LEGO Mindstroms NXT

The LEGO Mindstroms NXT, contains programmable arm processor called NXT brick. To this brick it is possible to connect three sensors and three motors through cables. Three of this LEGO Mindstroms kits were used in this project to build three different cars for each of the approaches respectively.

Each kit of NXT Mindstroms includes NXT microcontroller, Ultrasonic sensor, Colour sensor, Sound sensor, Touch sensor, Servo motors with built in rotation sensors, Connector cables and building blocks.

The essential parts used in construction of three robots are described below.

The NXT Brick

The NXT brick is the brain of the robot. It has I/O Ports to connect sensors and motors, a USB port for PC communication and Bluetooth for PC or inter robot communication.

Brick's specification:

- 32- bit ARM microprocessor.
- 256 KB Flash memory.
- 64 KB RAM.
- Bluetooth.
- 4 – Input Ports.
- 3 – Output Ports.
- LCD Display.

Ultrasonic sensor

The ultrasonic sensor transmits sound waves and receives reflected waves back from obstacle. It calculates the distance by half of the total distance travelled by sound waves.

Servo motors

The LEGO NXT servo motors have built in rotation sensors which calculate the speed of the motor and also calculate the total number of turned angle in degrees. The sign specifies the direction of the rotation. The rotation sensor has precision of 1 degree.

Camera

The camera has built-in embedded image processor which can able to track different colours and objects.

Laser

An ordinary laser light which can project a red dot on the surface on which the beam is pointed. The projected dot is then tracked by the camera. Totally 3 lasers are used in the car.

b. Softwares

The ROBOTC Programming Environment

ROBOTC is the premiere robotics programming language specially developed for LEGO NXT Mindstroms. ROBOTC is a C- based programming language. It's a separate platform developed for educational purpose and competitions. It contains separate firmware, so initially new firmware has to be flashed before using the brick.

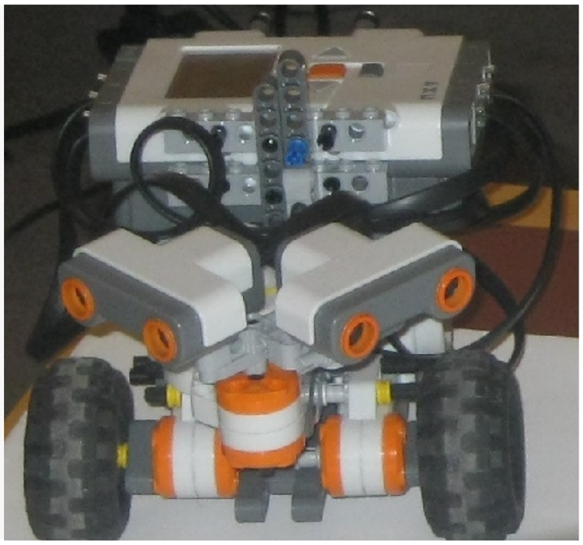
Software and drivers can be found at http://www.robotc.net/download/nxt/

6. Design and Implementation

a. Model 1

Design:

In this model two servo motors are used to drive the robot. In another servo motor, two ultrasonic sensors are placed at fixed angle to sense the surrounding environment. The servo motor which contains two ultrasonic sensors is placed in the front end of the vehicle. The motor allows the sonar sensor to rotate left and right giving the capability to scan the obstacles in a wider range. NXT brick is placed at back of the vehicle to maintain weight balance of the vehicle.



Algorithm:

The outline of the algorithm is that this modal scans the environment for any possible obstacles. If it senses any obstacle the car is turned to an angle such that the obstacle and the car are aligned parallel to each other. It uses only current sensor information. This algorithm contains two parallel tasks namely ControlCar and main tasks. It has modules namely Accelarate, TurnCar, updateOptimalSpeed, UpdateManeuveringDetails, decideMovement, TurnSonar.

- Task main:

  In this task the sonar sensor is turned 90 degrees left and back to initial position. This motion pattern is repeated periodically.

- Task ControlCar:

  The control task run in parallel with main program In this task, the two ultrasonic sensor values are obtained as the sonar keeps turning left and right periodically. Based on the distance value obtained the decision in the movement of the car is determined. This task is called for every 50 millisecond.

- TurnSonar()

  This function is used to rotate the ultrasonic sensor to the specified degree and in specified direction. The control signal for rotating sonar is generated and fed to the servo motor.

- TurnCar()

  The functionality of this module is to generate the control signal for driving the robot and it turns the car to desired angle based on the decision taken by the decideMovement module.

- DecideMovement ()

  According to the minimum obstacle distance the car movement will be decided so that robot maintains constant distance from the obstacle. This method decides if the car has to turn left or right or to go straight.

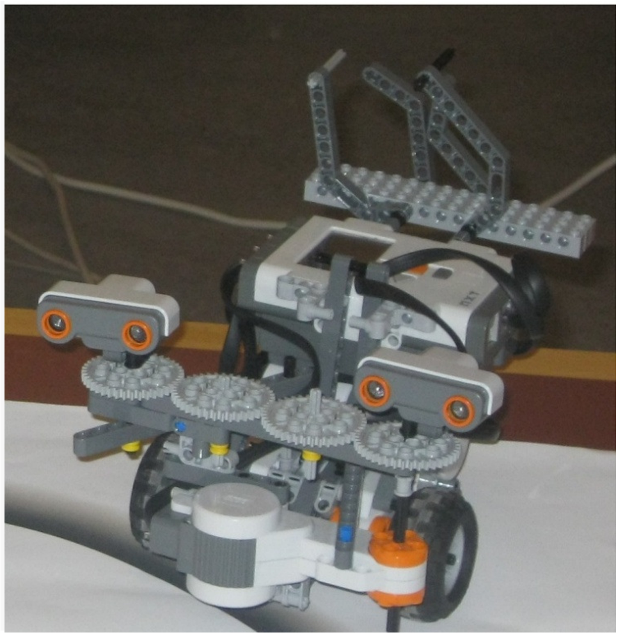
- UpdateSpeed()

  The speed of the vehicle is updated after every scan. The speed of the vehicle is set according to the distance of the nearest obstacle.

b. Model 2

Design:

In this model two ultrasonic sonar's are placed on top of the gear system which is connected to one of the servo motor. The gear system is built with four equal size gear rotators and one of gear rotator is connected to the servo motor. The ultrasonic sensors are placed at both the ends of the gear system and both sonar rotates in opposite direction. The servo motor which contains sonar is placed at the front of the vehicle and NXT brick is placed at back of the vehicle to maintain weight balance of the vehicle. Boundary following algorithm is implemented in this model.



Algorithm

This algorithm is based on boundary following algorithm as discussed in state of the art. The functionality for modal 1 and this modal are same except on how the sensor is turned for scanning and how the car turns when encountered with obstacles. In modal 1 the robot turns to an angle at which the sensor detects the obstacle. But this robot keeps turning gradually till it avoids the obstacle. This way it never deflects away from the obstacle but follows parallel to the boundary of the obstacle.

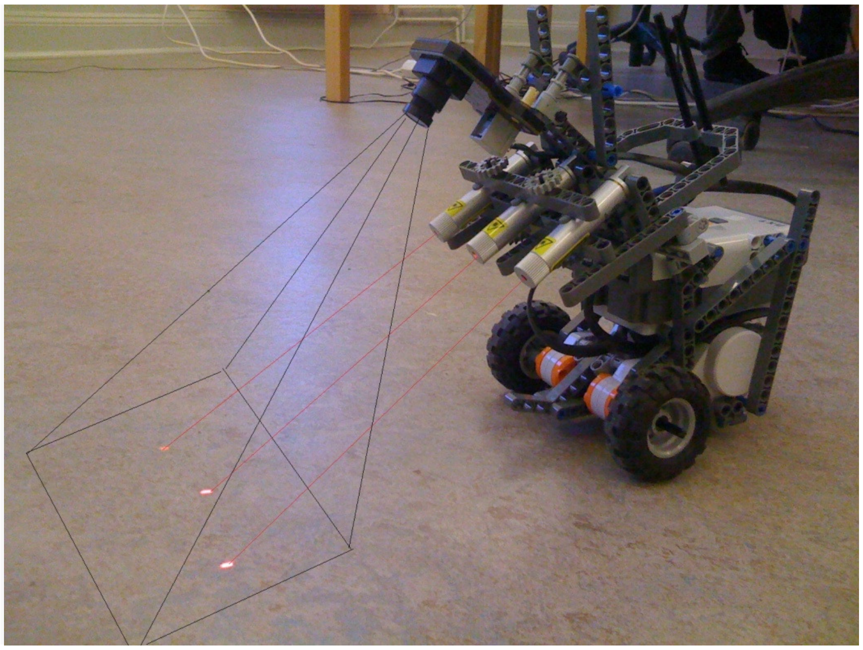
Pseudo code:

```
1. Initially the two sensors face straight towards front, as shown in the above
   picture.
2. Left sonar turns 45 degree left while right sensor turns to the same amount
   synchronously.
3. While the sonar turns, the obstacle distance is measured.
4. If any obstacle is encountered by left sensor, turn the car 20degrees right.
5. If the obstacle is encountered by right sensor, turn the car 20degrees left.
6. If both the sensors detect obstacle, keep turning to the direction in which the
   car was turning before.
7. Keep turning until it avoids obstacle and repeat the whole process indefinitely.
```

c. Model 3

Design:

In this model three lasers are placed at equidistance from each other on top of the robot. The camera sensor is placed such that all three pointed source of beam are covered in the view of the camera. This model is designed in such a way that placements of the field view of camera and the laser can be adjusted together nearby or far from the car. So the obstacle avoidance range can be approximately from 15 cm to 35cm.



Algorithm

The main idea in this obstacle detection technique is that, a row of laser beams are projected in front of the car. Those laser dots projected on the path are tracked by a camera. When the car moves the tracked lasers are always in a fixed location in the camera view, since both camera and laser moves along with the car. If the car encounters any obstacle, the laser dots are misaligned with respect to their initial location, as the laser dot will be projected on the obstacle rather on the floor. So this change in the location of lasers projection is monitored by the camera and appropriate decision is taken for cars movement, avoiding obstacle.

The camera sensor is capable of tracking objects of eight different colours. So the camera is preconfigured to track a bright red light source.

Pseudo code:

1. Initial locations of the three laser dots in the field view of camera are stored
2. Laser dots on the floor are tracked by the camera and its location in compared with the initial location
3. If there is no difference in the locations, car moves forward
4. If the Left laser dot is misaligned with respect to its initial location car turns right until the laser location matches with its initial location.

5. If the Right laser dot is misaligned with respect to its initial location car turns Left until the laser location matches with its initial location.
6. If middle or all the three laser dots are misaligned with respect to its initial location the car us turned to a direction previously car was turning.
7. Repeat from step 2

7. Controllers

a. Motor control for sensing system

This controller is implemented for model 1 and model 2 approaches which contains rotating sonar's mounted on the servo motor. By using system identification tool box in Matlab, the parameters of both models are identified and state space of the system is derived. The simple feedback controllers for both models are designed using automated PID tuning available in Mat lab. By using this feedback controller, the closed systems of both models are derived and simulation is done in Matlab. For the actual implementation, the PID values obtained from automated PID tuning will be sufficient, though it might not be precise. These values could be made accurate by tuning a little manually, by trial and error method. The code for controller is implemented in RobotC platform.

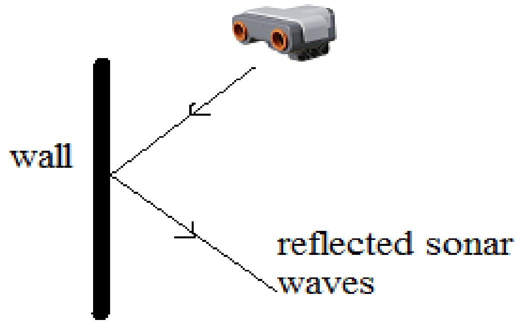
b. Motor control for driving system

Two different controllers are implemented for turning the car left or right direction. To turn the car both wheels rotate opposite to each other in synchronized way. The driving system is controlled by separate feedback controller for both servomotors. When the vehicle is turning left, right wheels servomotor feedbacks is used in the control system. When the vehicle is turning right, left servomotor feedbacks is used to control the vehicle. So in our implementation, always the wheel which has a positive voltage is considered for feedback.

By using system identification tool box, the parameters of all the models are identified and state space of the system is derived. In Matlab, by using SISO design tool box the simple feedback controller for both servomotors are designed. By using simple feedback controller, the closed loop systems for both servomotors are derived and simulation is done in Matlab. The controller code is implemented in RobotC is platform.

8. Limitations of sonar sensor

The environment of the robot is sensed through ultrasonic sensor. When the ultrasonic sensor is perpendicular to the direction of the obstacle, the sound waves will be reflected back by the obstacle and measured distance is reliable. However, this sensor is not reliable in all the scenarios. For instance when the sensor is not perpendicular to the obstacle, the sound waves are deflected away from sonar and hence will produce a wrong measurement.



Ultrasonic sensor fails to detect the obstacle in front.

Solution:

We know sonar is reliable only when obstacle and sonar are perpendicular to each other. So the sonar is made to rotate and senses for the obstacle in each degree. So this way irrespective of the alignment of the obstacle, it is possible to identify the obstacle by making the sensor perpendicular to the obstacle.

9. Conclusion and Results

Three different approaches were implemented and tested for collision avoidance. Compared to all the three approaches, modal 3 – laser and camera model was very robust in detecting and avoiding the obstacle. It was very precise and sensitive in detecting and avoiding obstacle. Even though other modals avoided obstacle they had many drawbacks such as

- The car does not turn to the desired angle, in case of smooth surfaces. Though wheel rotates precisely to make the car turn to desired angle, lack of friction on the surface makes the car skid and hence does not turn accurately. This could be overcome by having another controller whose feedback should be given by a gyroscope sensor.
- Not all kinds of obstacles are detected by sonar sensor. For instance obstacles which are very small and lies below the level of sonar sensor, mounted on the car. Obstacles which absorb sound waves and obstacles whose geometric shape makes the sound waves deflect away are impossible to detect by the sonar.

However the laser and camera sensors model works fine in daylight too, as the laser light is very bright. One constraint in this approach is that the distance at which the car should avoid obstacle should be fixed in the design itself. Even thought the obstacle avoidance distance can be adjusted, it should be done physically and not in the software.

When it comes to exploring areas modal 1 is more capable of exploring areas very quickly as it always deviates away in angle of alignment of the obstacles. This modal also comes out of a local minima situation. Whereas modal 2 and 3 adapts method more like boundary following technique. In future maps could be generated and stored so the decision on paths could be planned ahead.

10. References

[1] Michael Hoy, Andrey Savkin and Matt Garratt, *Navigation of an Unmanned Helicopter in Urban Environments* , 2010 11th Int. Conf. Control, Automation, Robotics and Vision Singapore, 7-10th December 2010.