Reactive Intelligence for Autonomous Row-Following Robot

Ethan Nutter, Dr. Santosh Pitla

Department of Biological Systems Engineering

University of Nebraska - Lincoln

Sharp Infrared Distance Sensors

0-5 DC volts into Arduino

Arduino Uno Microcontroller

Serial commands to Sabertooth

Sabertooth 2x25 Motor Driver

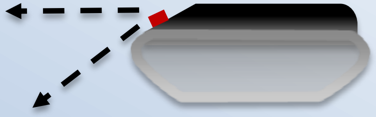
0-12 DC volts to motors

T’rex Tank Chassis

Hardware

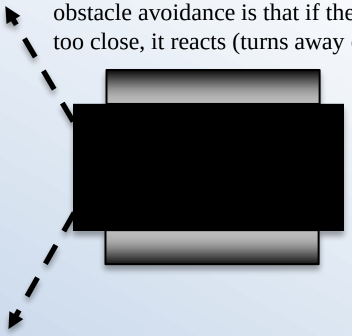
The robot’s body is a prebuilt tank chassis to handle rugged terrain. It’s brain is an Arduino Uno microcontroller which allows for simple reprogramming and quick prototyping. The bot’s muscles are two 4A 12V motors, which is much more current than the Arduino can provide, so a Sabertooth motor driver controls the motors using serial commands from the Arduino. Sharp infrared sensors serve as the robot’s eyes, allowing it to detect both the row it should follow and obstacles in its path. A 3S LiPo battery can power the robot for over an hour and a voltage regulator onboard the Arduino converts the 12 volts to 5 for the sensors.

How the Robot Sees

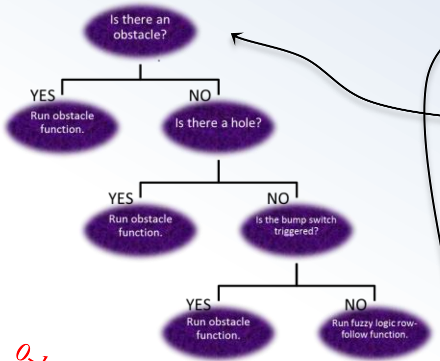


Angled-down front-facing sensors allow the robot to detect holes or edges (negative obstacles), and higher, level sensors allow it to detect solid impediments.

Side-facing sensors allow the robot to detect the row. The basic concept in following and obstacle avoidance is that if the robot gets too close, it reacts (turns away or stops).



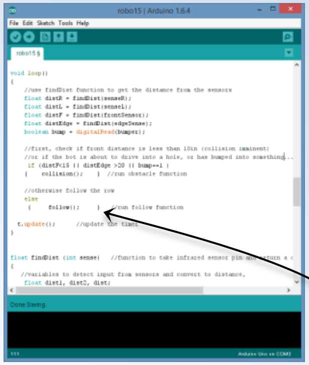
Simplified Navigation Logic



References

- SHARP GP2Y0A02YK0F datasheet
- Sabertooth 2x25 V2 datasheet
- Dimension Engineering Sabertooth Arduino library
- “Fuzzy Logic”, E. J. Mastacusa, Bucknell University
- “Autonomous robot obstacle avoidance using a fuzzy logic control scheme”, William Martin, CMU
- www.arduino.cc/en/Reference/HomePage

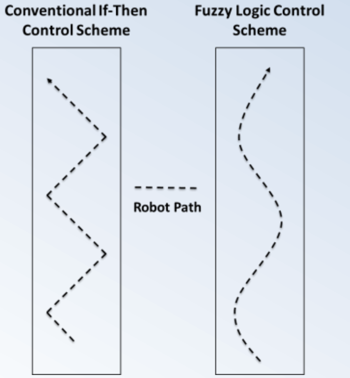
Software



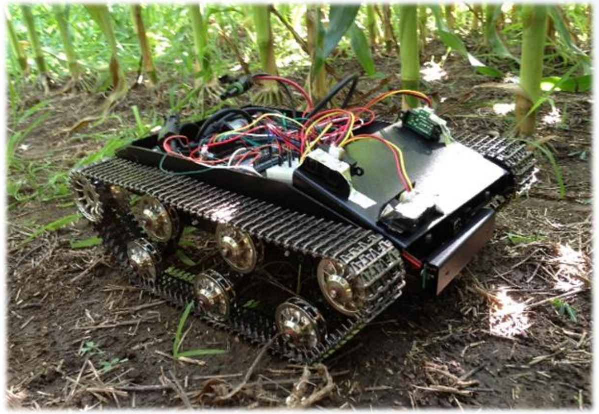
```
void loop()
{
   //use findDist function to get the distance from the sensors
   float distR = findDist(senseR);
   float distL = findDist(senseL);
   float distF = findDist(frontSensor);
   float distEdge = findDist(edgeSense);
   boolean bump = digitalRead(bumper);

   //first, check if front distance is less than 10in (collision imminent)
   //or if the bot is about to drive into a hole, or has bumped into something...
    if (distF<15 || distEdge >20 || bump==1 )
   {    collision();    }  //run obstacle function

   //otherwise follow the row
   else
    {     follow();     }  //run follow function

  t.update();      //update the timer
}


float findDist (int sense)   //function to take infrared sensor pin and return a c
{
  //variables to detect input from sensors and convert to distance,
  float dist1, dist2, dist;
```

The main portion of the Arduino code- the loop- runs repeatedly, instructing the robot to act differently depending on whether or not it detects an obstacle. Various functions are called on to perform a task such as convert the voltage from a sensor to a distance or make the robot follow the row.

Fuzzy logic allows for smoother more adaptive control and motion because instead of an object being ‘close’ or ‘far’, it can be ‘kind-of-close’ or ‘kind-of-far’. Moreover, instead of just ‘turn right’ or ‘turn left’, it is instructed to turn by degrees – ‘turn a little right’ or ‘turn a lot right’.



Conventional If-Then Control Scheme
Fuzzy Logic Control Scheme
Robot Path



Robot following corn row