Department of Electrical and Computer Engineering

EEL 5666

Intelligent Machine Design Laboratory

<u>S.L.I.K. 2001</u>

Salt Laying Ice Killer

FINAL REPORT

Daren Curry

April 22, 2001

Table of Contents

Abstract

The following paper describes the design of an autonomous robot that distributes salt where metal is present. The idea behind laying salt is that the target of the robot is iced over driveways. It is therefore reasonable to lay salt where metal if found because rebar is used in the construction of most driveways. The robot is being built using an existing toy bulldozer platform with a pull-behind hopper. The robot will be designed to spiral out looking for metal, spread salt while metal is detected, avoid all obstacles, and stop once the hopper is empty. The mobile platform will use a TJPRO11 board and sensors for feedback from the environment. S.L.I.K. will use IR detectors and emitters; bump sensors, and a pair of metal detectors for sensing metal. The motivation behind the design of this robot was derived from visiting a dear friend who lives in Indiana. In the mythical state of Indiana, it is not uncommon for white, flaky stuff to fall in the winter, and due to this and the cold weather, ice forms everywhere. Moreover, while visiting this dear friend of mine, it was necessary to get a running start at his driveway in order to assure reaching the house, since the driveway was rather steep. After realizing how dangerous this was, I enquired about how to fix the problem I was given a bag of salt and made to throw it in the cold. The use of this robot will keep driveways from icing up and keep me from freezing my ass off.

EXECUTIVE SUMMARY

S.L.I.K. is a tracked autonomous robot. Its sole purpose is to seek out metal and throw salt in an effort to de-ice driveways. If it doesn't find metal, and both servos have reached their maximum speed, S.L.I.K. reinitiates the spiral pattern so he does not continue off in a straight line. If metal is found, the hopper is activated until the metal is no longer present or the hopper runs out. If the hopper runs out S.L.I.K. stops and waits to be filled and reset. Lastly if an obstacle is encountered, S.L.I.K. takes appropriate actions to avoid objects. S.L.I.K. consists of a preexisting toy bulldozer platform made of plastic and a hand-held hopper modified to be pulled behind S.L.I.K. The main challenge of this project is to determine the presence of metal. The other challenge was to program the correct timing and speed. The fully integrated platform was achieved by using the TJPRO11 board, Infrared sensors, bump sensors, and metal detectors. The designed sensor was the metal detectors used to determine if metal is present. The hopper was used to hold and distribute the salt. The infrared sensors were utilized in the object avoidance. The contact switches were used as a redundant system for obstacle avoidance, and were used to determine if the hopper was empty. The tracks are actuated by to "hacked" servos.

Introduction:

Going up north and watching people throwing salt and having the misfortune to do it my self in the cold, made me realize that there had to be a better way. This led to the creation of S.L.I.K. Many places suffer from cold weather, and could appreciate the value of a robot that salted driveways. However, since the beginning of this project, it has occurred to me that with slight modifications, S.L.I.K. could be used to spread fertilizers and such in a yard. Since S.L.I.K. is a tracked vehicle, the only major physical modifications lay in the hopper. I have tested S.L.I.K. in grass of moderate height and encountered no problems with locomotion. A small, inexpensive autonomous robot that could perform these and other repetitive tasks is worth pursuing. The objective of this project is to build an autonomous robot that will distribute material when metal is found. This objective will be met using a variety of sensors, servos and motors. The paper will discuss the entire integrated system, following with servos and the sensors used to accomplish the objective. Finally, I will discuss behavior algorithms and the specific code used to achieve the required objective.

Integrated System:

The completed system will consist of two metal detectors for metal finding, two ir emitters and receivers for obstacle avoidance, three contact switches, two for backup to the ir and one for the hopper empty detection. These two servos are connected to PA4 and PA5 on the TJPRO 11 board. The integrated system will be powered by six AA Nickel-metal Hydride batteries (for the micro-controller), four AA Alkaline batteries (for the hopper), and two 9 Volt batteries (for the metal detectors).   The Infrared emitters are powered by a 40 KHz signal generated by the PE2 and PE3 port on the TJPRO 11 board. The front left and right contact switches will be connected to the FBRSW and FBLSW port respectively on the TJPRO11 board. The hopper empty switch will be connected to the RBSW port on the TJPRO11 board.   Motion will be accomplished by two servos. These two servos are connected to PA3 and PA7 on the TJPRO 11 board.

Mobile Platform:

The mobile platform consists of a pre-existing toy platform that contains Ir detectors and contact switches for object avoidance, wheels attached to hacked servos for locomotion, and two metal detectors, and a pull-behind hopper. The Robot will travel in a spiral pattern until it bumps into metal or an obstacle. Once the robot has found metal, the hopper will be triggered, and the robot will go straight for a predetermined amount of time. If a non-metallic object is encountered the avoidance routines will be executed. The most significant problem was the interference of the metal detectors with themselves. To combat this problem, I placed ferrite material in between them to isolate them. The only problem with this was the creation of a “blind-spot” in the middle of the sensors. Since this introduced a blind spot, the presence of metal is not assured. The robot might not know if it has just encountered metal or if it has already been encountered. However, there is a solution to this problem. To fix this problem is to add a third metal detector that would determine if the robot is still on the line or not. Unfortunately, no metal detectors were available at radio shack, and that project was post-poned till the summer.

<u>Sensors:</u>

I attached most sensors to the TJPRO11 board directly. Since, the code for reading the sensors has already been written by Professor Doty, the made the job of coding much easier. My design called for seven sensors: two IR sensors, three contact switches, and two metal detectors.

IR:

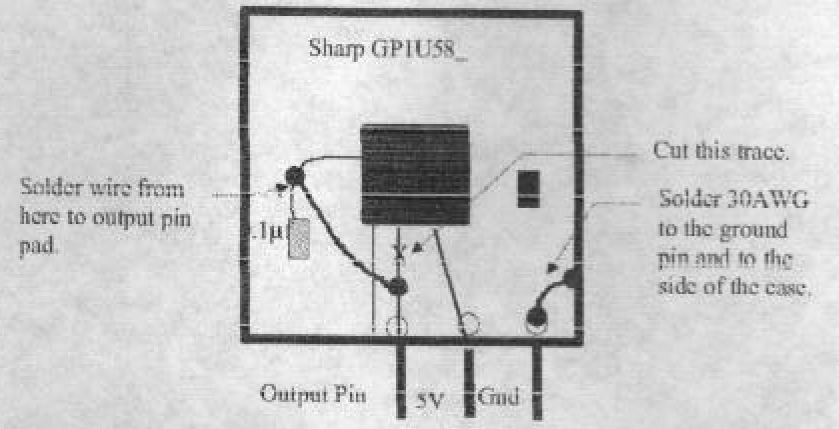
Sharp GP1U58
Cut this trace.
Solder wire from here to output pin pad.
.1µ
X
Solder 30AWG to the ground pin and to the side of the case.
Output Pin
5V
Gnd

Figure 1 Converting a digital IR sensor to an analog IR sensor requires cutting the trace to the Output pin, soldering the Gnd pin to the side of the case, and connecting the output of the 0.1 µf capacitor to the Output pin.

Figure 1

Figure 1 above, depicts the hacked version of the Sharp GP1U58 IR detector that I used for obstacle avoidance. This allowed me to determine approximately, how far away S.L.I.K. was from an object or person. The Ir was placed on top of the plow (see figure 2).

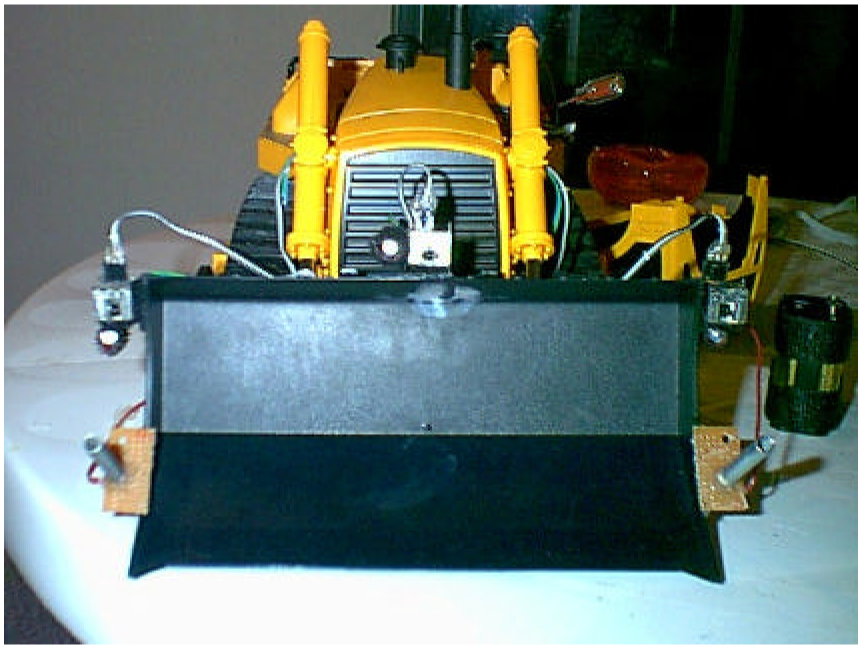

Figure 2

Further, when the port PE2 or PE3 got a reading of 127, the robot was approximately 3 inches away from the object. Some factors determine the capability of the detectors. For example, if it was a dark wall in front of the robot, the robot could not detect.

Bump:

The bump sensors consist of 3 contact switches, 2 mounted at the front of the platform, and 1 in the hopper (see figures 2 and 3). The front 2 are used when the robot is moving forward, and as a backup to the ire. If the bumper is getting activated an obstacle has been hit and the robot will go into the avoidance routines. If the rear bumper was pressed, that means the hopper is full. If the rear bumper is not pressed, the hopper is empty and the robot stops.

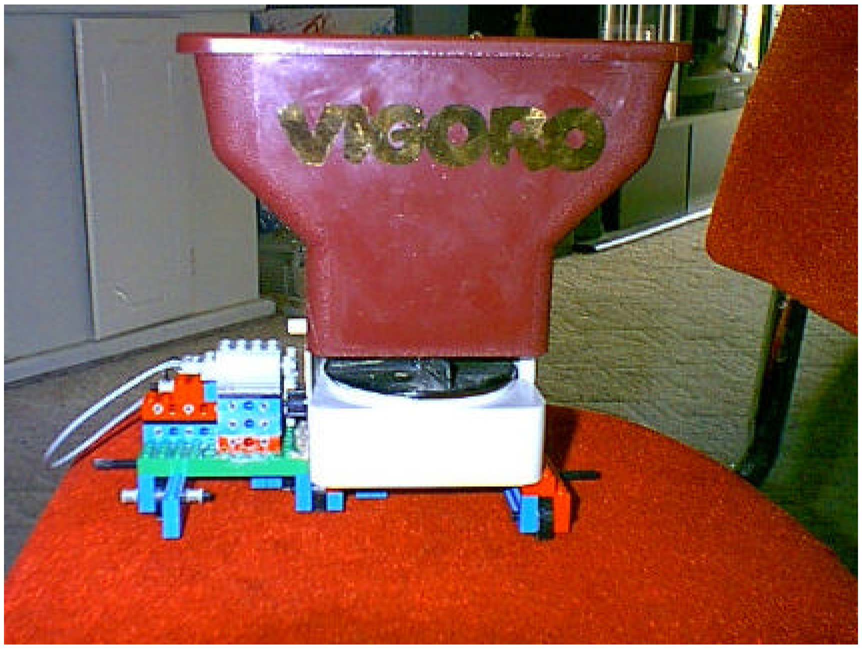

Figure 3

Metal Detectors:

The metal detectors are located in the bottom of the plow (see figure 4). The metal detectors are used to find metal. The input to the micro controller was done by gluing a CDS cell to the stock diode on the metal detectors. If metal is present, the ports read over 230, and under 190 if metal is not present.

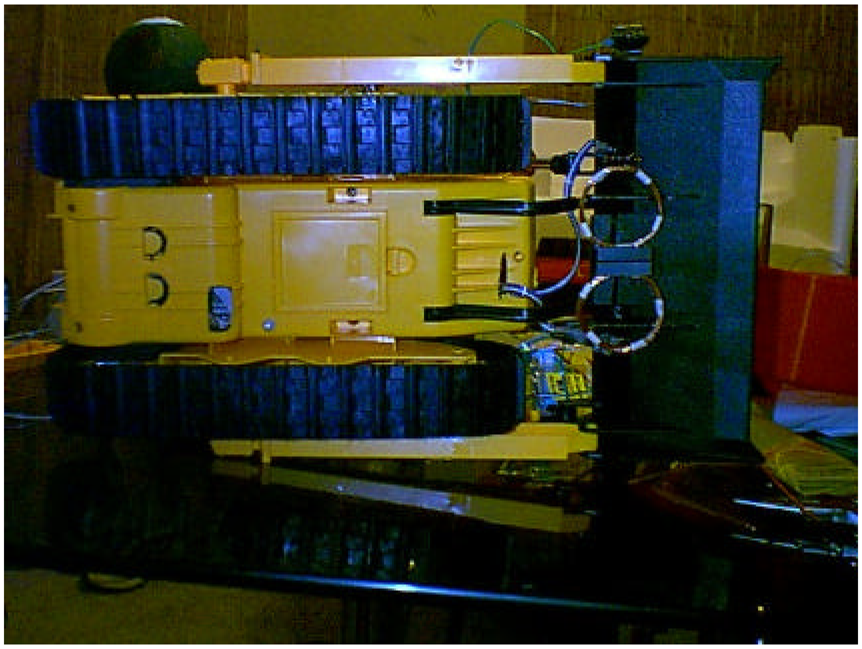
Figure 4

Actuation:
The locomotion of S.L.I.K. was provided through two hacked servos. The servos were mounted by removing the factory drive train, and rigidly mounting the servos to the body of the robot (see figure 5 and 6).

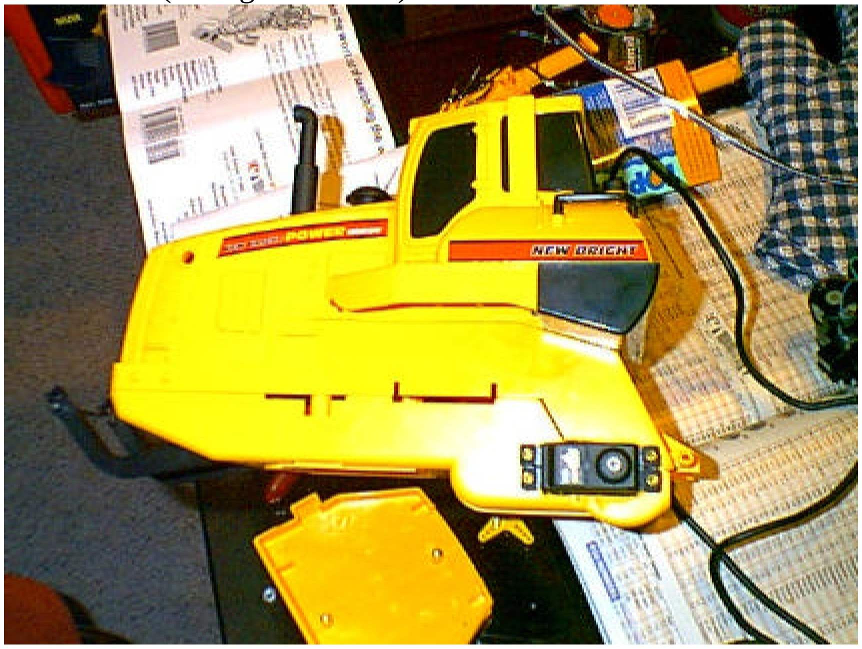

Figure 5

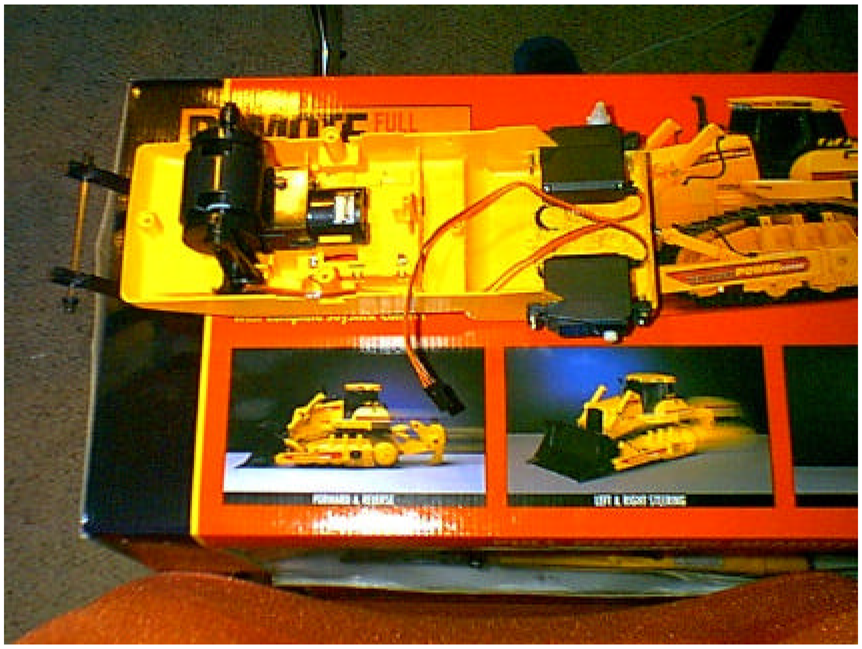

Figure 6

The only problem with this was the tendency to hop slightly. However, the robot did go fairly straight, especially since no feedback exists on the platform.

Behaviors:

The four behaviors that will be demonstrated are search, metal detection, avoidance, and hopper empty. The initial behavior is the search. Upon hitting the reset button, S.L.I.K. will start in a small spiral, and gradually increasing one servo in order to increase the diameter of the circle. The only problem with this is that in small circles it repeats several times, but with out this delay, the spiral would not be complete when the motors are close to the maximum speed. The pattern is also reset after the maximum value for the servo is reached. The next behavior is metal detection. In this behavior, while metal is detected, the hopper will be activated, and S.L.I.K. will move off in a straight line. Once the metal is no longer present, the spiral pattern will be resumed. The avoidance is a sub-function within the prior two. If at anytime an object is encountered, the avoidance routine is called. The avoidance routine reacts differently depending on what sensor senses the object. If a front ir detects the presence of an object, the robot will turn away from it. If the front bump switches are hit, the robot backs up and turns to the right. Lastly, if the hopper empty condition arises S.L.I.K. simply stops and waits to be refilled and reset.

Conclusion:
I believe that my robot has potential. Even though the original goals of line following were not achieved, I think that with some modifications to the existing platform will allow for the addition of line following. I also think that it could be a good proof of concept robot.

Code:

```
/*************************** Includes ********************************/

#include <tjpbase.h>
#include <stdio.h>
#include <hc11.h>

/************************ End of Includes ****************************/


/*************************** Constants
********************************/

#define AVOID_THRESHOLD 100
#define Metal_detect 230

/************************ End of Constants
****************************/


void main(void)
/****************************** Main
***********************************/
{

    int count , inc , speedr , speedl, check;
    check = 0;
    count = 0;
      inc   = 0;
      init_analog();
    init_motortjp();
    init_clocktjp();

  IRE_ON;     /* turn on IR emitters */



while (BUMPER > 120)
{
/********************************* spiral
routine************************/
      if (check == 0)
      {
            speedl = MAX_SPEED;
      speedr = 60;
      check  = check + 1;
    }
      else
      {
      inc    = check/350;
      check  = check + 1;
      speedl = MAX_SPEED;
      speedr = 60 + inc;
      }
      motorp(LEFT_MOTOR, speedl);
      motorp(RIGHT_MOTOR, speedr);
      wait(30);
/****************************** find metal/ turn on
hopper**************/

            if (RIGHT_MD > Metal_detect)
      {
            DDRD = 0x10;
                  PORTD = 0x10;
                  motorp(RIGHT_MOTOR, MAX_SPEED);
            motorp(LEFT_MOTOR, MAX_SPEED);
            wait (2250);
      }
      else
            {
             DDRD = 0x10;
             PORTD = 0x00 ;
            }

            if (LEFT_MD > Metal_detect)
            {

            DDRD = 0x10;
                  PORTD = 0x10;
                  wait (1500);
                  motorp(RIGHT_MOTOR, MAX_SPEED);
            motorp(LEFT_MOTOR, MAX_SPEED);
            wait (2250);
        }

        else
            {
             DDRD = 0x10;
             PORTD = 0x00 ;
            }

/************************************ avoid
*********************************/
         if (LEFT_IR > AVOID_THRESHOLD)
          { motorp(RIGHT_MOTOR, -MAX_SPEED);
            motorp(LEFT_MOTOR, MAX_SPEED);
            wait(450);
          }

            if (RIGHT_IR > AVOID_THRESHOLD)
             {
               motorp(RIGHT_MOTOR, MAX_SPEED);
               motorp(LEFT_MOTOR, -MAX_SPEED);
               wait(450);
             }

             if(FRONT_BUMP)
              {
                motorp(LEFT_MOTOR, -MAX_SPEED);
                motorp(RIGHT_MOTOR, -MAX_SPEED);
                wait(600);
              }
              if (inc== 40)
              {
              check=0;
              }

 }/*END WHILE (BUMPER <120)*/
 /************************************ hopper empty
*********************/

 while (BUMPER < 120)
      {
         motorp(LEFT_MOTOR, 0);
       motorp(RIGHT_MOTOR, 0);
       wait(600);
    }
}
```