Glitchy

A Hardware and Software Approach

Design Group 3

Clint Ayler

Jonathan Cross

Nathaniel Dale

Greg Zolas

Table of Contents

Abstract

"Glitchy" the robot is built upon a modified Ackerman drive mechanism. Not only do the front wheels steer, but the rear axle is capable of assisting as well in order to negotiate tight turns. The design is a hybrid, using an HC11 and an Altera chip to interface with and control motors and electronics borrowed from radio controlled cars. Intended to compete in the Trinity competition (http://www.trincoll.edu/~robot/), we sought to maximize its advantages at each level of its design.

Introduction

In our Jr. Design course, we were faced with a choice between repeating a project that has been done for several years running, or defining and attempting to carry out a brand new project unique to our group. We chose both.

The robot we created is known as "Glitchy" after a certain tendancy its motor control aquired and managed to keep during most of its developmental stages. It's the product of an unflinching design philosophy that takes a fresh approach to solving old problems. Wherever feasable, we parted from more traditional design solutions in order to persue new ideas, sometimes coming up with innovative designs that worked. Often in the end, we did end up doing what is normally done, but only when we were convinced it was the most appropriate choice after exhausting the alternatives.

Following is a detailed description of what we did and why, starting from the nuts and bolts of the mechanical aspects, then progressing up to electronics and finally software.

1 Chassis Configuration

Our main goal in choosing a chassis and drive mechanism was to create a unique and new design that was also practical. While the Differential Drive robots are highly maneuverable, we wanted to avoid their many drawbacks such as instability on uneven surfaces, redundant motors and motor control electronics, and programming complexity. We settled upon a modified Ackerman chassis consisting not only of steering front wheels, but also an articulated rear section to achieve a tight turning radius.

1.1 Layout

The chassis consisted of three main layers. From lowest to highest it was mechanical, power, then electronics.

Since the base would be subject to the most amount of mechanical strain, we chose a .080 thick sheet of aluminum of size 6” by 8”. Large all-thread stays were attached for support of the upper two levels, and the mechanical steering mechanisms were located on this, the smallest level.

For the upper layers, we chose 3/16” thick transparent Acrylic sheeting because of its availability and weight savings. The second layer was a 8½” by 8” sheet, and housed the batteries. This level also had attachment points for the base stays allowing the second and third levels to be quickly and easily disconnected as one piece should modification or maintenance be required to either section.

The top, being clear of all obstructions, was the largest at 8½” by 11”. This is where all the electronics were located.

1.2 Mechanics

On the bottom level, we employed the use of a small door hinge to attach a differential axle to the robot. The advantages to using the differential are: 1) With a mechanical differential, just a single motor can be used yet both wheels receive power. With this single motor, gear ratios can be tailored to give a specific torque or speed desired, allowing slow speeds smoothly even with large tires. 2) It provides an easily steered rear section to manipulate. 3) The availability and cost of one was such that we could not pass it by.

In order to actuate our steering, we utilized a servo which is mounted near mid-chassis with metal bars attached to the front steering mechanism and the rear differential section.
The front wheel hinges were placed in custom aluminum mounts, which had to be fabricated in order to adapt the steering mechanism to our frame.

Also on this layer can be found the white line sensor position in a cutout approximately 2" deep by ½" in the very center of the front of the chassis. For an overall view of the first level see Fig 1.1

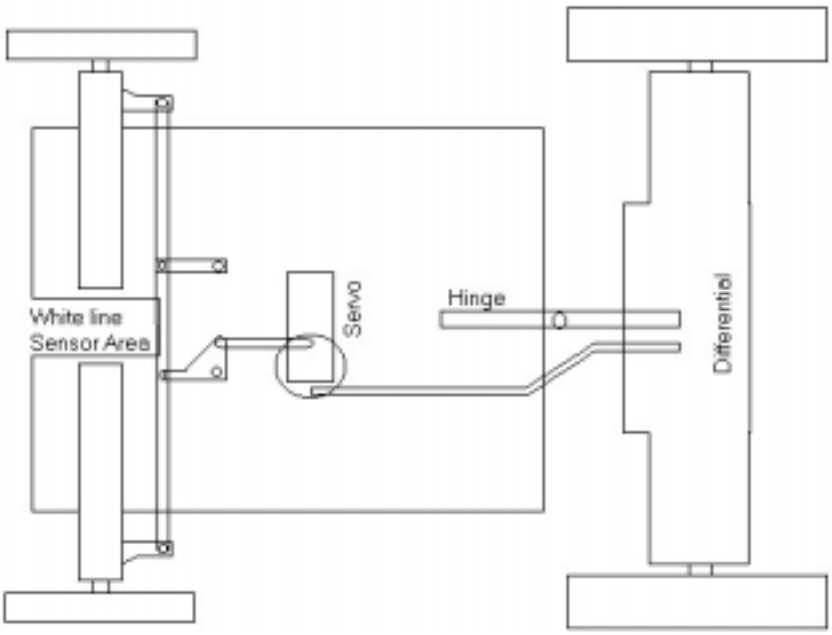

Fig 1.1 – First Layer General Layout

On the second level the entire power storage of the robot can be found. The acrylic sheet had a section of the right rear removed to accommodate the swing the motor during right turns. (See Fig 1.2). The remaining portion had enough space to accommodate our `12V 2.3 Ah Ni-Cad battery along with three other 7.2 volt batteries to power the motor. Stabilization of the batteries was a problem at first. Using frictional methods of attaching the batteries along with a high density foam pressed down over the batteries and secured tightly by the position of the top layer, effective battery containment was achieved.

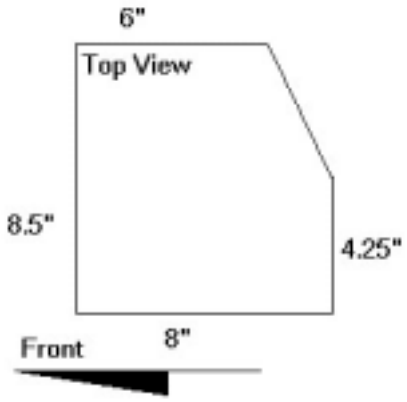

Fig 1.2 Top View of the second layer.

The third and final layer consisted of the HC11 and all the wire-trace boards created for the robot. Each board was attached to the acrylic through the use of 1" spacers to raise the boards above the layer and provide room for the running of wires and connectors. On the two front corners of the layer, the GP2D12 wall sensors are hung on right angle brackets at an outward angle of 45º.

Our plans were to include a raised platform for our fire extinguishing system to be mounted upon. This system consisted of a model airplane propeller mounted backwards on a dedicated motor. In our tests, we were able to extinguish a candle at distances greater than 6 feet, which was more than adequate.

2. Hardware Design and Implementation

Due to the unique nature of the chassis design the hardware design is also quite unique. We relied heavily upon components designed for radio controlled (RC) cars because they tend to be built to very high specifications regarding durability. It also helped that we had almost $400 worth of said components on hand and ready to use.

2.1 Power Storage

The system utilizes two separate power sources; one 12V and the other 24V. The 12V source is a 2.3Ah lead acid battery commonly used in video cameras and is used to power the electronics. The 24 V source is a series of three 6-cell (7.2V) rechargeable Nicad battery packs commonly used in Radio Controlled (RC) Car Racing and is used to power the drive motor H-bridge, the fan motor H-bridge, and the two servos. The 12V circuit powers everything else in the robot, including the HC11 microcontroller, Altera PLD, sensors, and other interface hardware.

2.2 Power and Regulation

The 12V battery is run into two separate 5V regulators, one used for Analog circuitry and the other for digital circuitry. The same regulators are used for both circuits. We chose the Linear Technology LM2940's due to their current capacity of 800mA and its low dropout voltage of 1V. We heat sinked the digital Vreg because it was sourcing twice the current and we sought to avoid heat problems. On each we used the recommended capacitors on both input and output in an attempt to keep the noise level down. Also, a low current 6V circuit was required for the sole purpose of driving the PWM control lines to the H-bridges.

2.3 Motor Control Hardware

The drive motor H-bridge is an interesting package. It is manufactured by Novac Electronics, and is a very tough unit. Its original purpose was to control high output radio controlled racing motors, motors that are more than capable of drawing 50 Amps continuously and withstand

current spikes of over 200 Amps at voltages varying between 12V and 32V. Because the H-bridge was intended for racing, it was thought to be a good choice to control the Pitman motor being that it was found to draw less than one amp at 24V under heavy loads, well under the normal operating conditions the H-bridge was intended for.

In order to use this H-bridge it had to be reverse engineered to a certain extent. We found that this unit output a fully-proportional pulse width modulation signal at 3.2kHz and required an input pulse width modulation (PWM) signal of 50Hz whose duty cycle ranged from 5 to 10%. Also, it was found that the H-bridge triggered at about 4.8V. This initially caused problems with glitching because there was a great deal of noise at the trigger level due to the digital circuitry, so it was determined that the input voltage should be raised to 6V in order for the H-bridge to run properly. After this small adjustment the H-bridge worked flawlessly and proved to be nearly fail proof, as it never malfunctioned despite all of the testing it went through. The second H-bridge used to drive the fan motor was of similar design manufactured by Teakin Electronics. Also intended for use in Radio controlled cars, it was not as heavy duty as the Novac being that it was intended for lower torque motors but it still exceeded requirements. One other nice feature of both of these H-bridges is that they have a built-in 5V regulator to power external electronics like the servos which allows them to run using the 24V supply circuit without requiring a separate power circuit.

2.4 Servo Motors

The H-bridges previously discussed were designed to run directly off of the same signal as the servos, also borrowed form the Radio Controlled world. We used two different servos; one was a high speed, high torque servo made by Arotronics Corporation, while the other was a standard generic servo made by Cirrus. The Arotronics servo has approximately 75 oz-in of torque with a reaction time of .45 sec to move through its full range of 180º and is used to actuate the steering mechanics of the robot. The other servo is rated at 42 oz-in with a reaction time of .64 sec to turn through 180º. This servo is used to rotate the fire sensor once a room is entered.

2.5 DC Motors

The 24V circuit drives the motors used to move the robot and to put out the flame. The motor used to drive the robot, previously mentioned, was manufactured by Pitman. It has a maximum voltage of 24V and produces a torque of 175 oz-in with a maximum speed of 1000 RPM after a 5.9:1 gear reduction. Also included in the Pitman motor package are two encoders that are slightly out of phase with one another, each having a resolution of 500 pulses per rotation. The encoders require an input voltage of 5V and output a good square wave signal that can be fed into digital circuitry with little trouble or need for external circuitry. The other motor was intended for nothing but brute force; as it is a "mild" RC motor it only draws 10 Amps under load. It is a simple DC motor without encoders or gear reduction, but produces an estimated 1000 oz-in of torque at 28,000 RPM, and with a reversed RC airplane propeller attached it could extinguish a candle more 6 feet away, if necessary.

2.6 Motor Direction

The last item connected to the digital Vreg was the MOSFET that actuated the double pull, double throw relay, which controlled the motor speed direction by reversing the polarity on the motor leads. The control line for this FET is one that the Altera chip controls, so, by setting a single bit in the motor speed register, the direction could be changed. This was necessary because the "H-bridges" I spoke of earlier are not true H-bridges but are essentially 10 MOSFETS connected in parallel with external circuitry required to perform operations described earlier in Section 2.2, and so they are only capable of turning the motor in one direction. The one output line from the Altera into the HC11 is held high when new information can be written to the Altera.

3. Analog Circuitry and Sensors

Unlike the digital circuitry, there are few components that comprise the analog circuitry. With the exception of providing a reference voltage to the HC11's A/D converter, the analog circuitry is only used for powering the sensors. In choosing our sensors, we placed emphasis on reliability and consistency since the robot's sensors are the most subjected to varying conditions such as ambient lighting, differences in paint color, and dirt.

3.1 Distance Sensors

For distance measurements, we chose the Sharp model GP2D12, a self contained emitter/detector triangulating pair with all the necessary driving circuitry in a small, but complete, package. This type of range sensor was easy to use because it needed only to be connected to power and ground, and its output is analog. We buffered the output to try to cut down on noise before going to the HC11 for A/D conversion. While it proved to be quite invariant to differences in wall colors and lighting, it was not without its drawbacks. The distance to voltage response was not a one-to-one correspondence, but that could be dealt with in software. The main drawback of this sensor was that it had a limited range of 40 cm. It also required a minimum range of 10 cm, cutting its usable region down further.

3.2 Fire Sensor

The fire sensor design was basic but practical. It consisted of an infra red phototransistor mounted inside a reflective light guide tube. The analog output is then buffered, and passed to the HC11 for A/D conversion.

The phototransistor, which was had a peak response in the near infra red spectrum, also responded to ambient light. This problem was amended by the use of the light guide tube, a clear acrylic tube of 3/4" diameter which was wrapped with reflective copper shielding tape. The open end of the tube was covered with a piece of film from a static sensitive bag which acted as a filter, removing much of the ambient visible light while still allowing much of the near infrared

from a flame source pass. The length of the tube could also be cut to length so as to give a larger or smaller field of view for the phototransistor in order to resolve and collimate light radiated from the candle. The fire sensor was mounted neatly on a small etched circuit, ready as a whole to be scanned over a room by pivoting it on top a servo. With the configuration shown below a full response of 5V would be observed by the sensor when a candle flame was within a few inches of the sensor and varied inversely proportional to the candle's distance, with it still giving a 2V response to the candle at more than 3 feet.

3.3 White Line Sensor

Similar in construction to the fire sensor, the white line sensor uses a phototransistor mounted in a light guide tube to respond to near infrared radiation. This is coupled with an infrared emitter that is mounted in tandem with its own guide tube. The final output from the phototransistor is then Schmidt-triggered so that a difference in response between black and white will correspond to 0 and 5 volts, respectively, giving a digital logic-level output which can then go directly into an input line of the HC11.

The 1 inch long light guides were made of ¼" inner diameter clear tygon tubing wrapped with copper tape and slid over the LED and phototransistor. As shown, the IR LED's radiation is channeled along the light guide towards the floor where it is then partially reflected or absorbed. The purpose of the light guide sheathing around the phototransistor was to block light interference from other sources while still allowing reflected IR light to reach it and be detected.

3.4 Audio Tone Decoder

In the Trinity College Firefighting Robot Competition, it is required that we use a single-frequency audio tone of 3.4kHZ to signal to the robot that it must start its way through the maze. Necessary for this is a microphone followed by an amplifier and then some means of selectively giving a response depending on whether the frequency detected falls within some frequency band centered around 3.4kHz.

Used for this sensor was a dynamic (magnetic coil) microphone taken from a set of Sony walkman headphones which required no driving voltage for its operation. The output of this microphone was then amplified before going into an LM567 tone decoder produced by National Semiconductor. This tone decoder could be adjusted via external circuitry to control the passband width and center frequency; its output would change from a high of 5V to 0V when a frequency within the passband was detected.

4. Digital Electronics

Connected to the 5V digital regulator were several components Including the HC11 EVBU, an Altera PLD, a decade divider whose purpose will be explained shortly, and the buffer circuitry required to interface the HC11 and Altera to the H-bridges and servos.

4.1 HC11 Microcontroller

We chose the Motorola HC11 Microcontroller for a number of reasons. First, we have been using it for sometime now and are familiar with its functionality, interfacing and programming. Second, additional memory has previously been added, allowing a larger (better) program to be installed. Lastly, between the four of us we have 4 HC11 EVBU's, three of them with the memory expansion. So, if one were to die on us, it could be easily replaced. Several of the HC11's functions were very useful. The 8-bit pulse accumulator was used for motor speed calculation. In order to do this we put the output from one of the motor encoders into a decade divider. This decade divider, as its name suggests, would output one pulse for every ten it received. This output was then put into the pulse accumulator of the HC11, allowing the motor speed to be calculated with 8-bit resolution. Another nice feature was the built in serial communications port that allowed it to be monitored and programmed via a PC terminal.

4.2 HC11 I/O

Initially we were worried about bogging the HC11 down if we used the peripheral interface adapter (PIA) ports added with the memory, especially with the pulse width modulation signal

needed for the H-bridges and servos. So our goal was to keep the data and address lines that the expanded memory uses free by not using the PIA, but this severely limited the I/O capabilities of the HC11. To compensate for this we decided to implement the PWM signal using an Altera chip. In order to interface to theHC11 to the Altera we used eight output lines and one input line. Four of the output lines were data lines dedicated specifically for information transfer between the HC11 and the Altera. Two of the eight outputs were the function selection lines that acted to select which of the PWM signal we were going to modify: motor speed and direction, steering servo and fire sensor servo angle, and extinguisher motor on/off. Since eight-bit resolution was necessary and we only had room for four data lines, one line was used as an upper and lower address select. So information was passed one nibble at a time, each function requiring two nibbles. Lastly, the Altera chip required a clock signal.

4.3 Noise and Transient Issues

It was realized that clock signal from the HC11 E-clock going to the Altera chip was a little noisy, so the clock signal was made to run through a Schmitt trigger. The output of the Schmitt trigger then went to the Altera chip, providing a much cleaner signal.

As with the use of the Schmitt trigger on the E-clock signal, it was determined that the output of the Altera should be buffered in the event that one of the H-bridges or servos shorted out or drew too much current. So, being that we had four signals that we were concerned with, we decided that an OR gate, with a slightly higher input voltage of six volts and one lead tied to ground, would work well. As mentioned earlier in Section 2.2, the six-volt input was necessary to ensure that the H-bridges would trigger properly.

4.4 Altera Programming

As mentioned earlier in Section 4.2, we felt that that having the HC11 perform the PWM would use up too many of its I/O pins, so an Altera PLD was used for this purpose. The Altera chip has five functions: speed control, motor direction, steering servo angle, fire sensor angle, and fire extinguisher motor control. These signals, except for the motor direction signal, are PWM

signals. As was explained earlier, the data for each signal is sent in two nibbles. The data sent is saved in a different latch for each function. The motor direction control is a simple high or low output depending on the control bit in the drive motor register. The PWM signals are produced by first creating a counter. This counter had to be large enough to create a 50Hz PWM using a 2MHz clock, which would produce forty-thousand E-clock pulses requiring a 16-bit counter. Since our PWM varied between 5% and 10% that means that, regardless, the first 5% of the duty signal is high and the last 90% is low only leaving 5% to be adjusted. Some 5% of the signal is equivalent to approximately two thousand pulses. So all that is needed to be done is to compare the current clock setting to the desired specifications using an IF statement. If the clock value is less than 2000 then the output is high, and if the clock value is greater than 4000 the signal is low. The tricky part is to correctly control the area between 5% and 10%. To do this you need to compare the clock to the value stored in the latch. This can't be done directly--in order to get the desired results you must compare the value of the counter (minus the initial number of counts) to the shifted version of the latched value. Also, the appropriate number of 0's needs to be tacked onto the front of these values so that two 16-bit numbers will be compared. Then, the counter is reset when it reaches the final value necessary to make the appropriate base frequency. When the IF statement is true the PWM line is set high and when it is false the line is set low. All of the PWM signals work in this manner, but vary slightly depending on specific needs. (See the Altera code in the Appendix.)

5. Software Approach

Efficiency and flexibility were the key goals in developing our main program algorithm. Since our chassis style was untried, changes in the mechanical structure were inevitable as the design progressed. Time was not a luxury we had, so we could not afford to wait until the end before coding started. Thus, a program had to be written that could be readily adjusted and changed frequently. We also hoped our unique drive mechanism would allow us to move much more quickly than an equivalent differential drive robot, and sought to maximize the speed and efficiency of the HC11 so it could keep up in real time.

5.1 One Short Interrupt

A major concern was the prospect of a missed interrupt or a timing conflict. Instead of implementing multiple interrupts and input captures, we chose an algorithm (see Appendix for flowchart) that utilized a WHILE loop that would continuously scan and update Boolean values. Only one interrupt was used out of necessity, the RTI in order to be able to consistently make time based measurements such as distance and velocity. Even so, it was kept extremely brief, updating and incrementing just a handful of variables before returning to the main program. The initial interrupt setting of 32 Hz turned out to be more than fast enough, but the option of increasing it to 64 Hz was available. The main program was tested and measured to loop more than three times per interrupt, leaving plenty of room for program expansion even at the higher RTI rate.

5.2 Integer Only Calculations

The most significant factor in the speed at which our program loop ran was that we used all integer mathematics. Since the HC11 contains no floating point coprocessor, we stayed strictly with integer representation. In some cases, we had to multiply up to keep precision in our calculations, then divide back down. But even these extra steps are significantly faster than attempting even simple floating point operations. Nothing was sacrificed by avoiding floating point variables. In fact, the extra multiplication steps of scaling were often unnecessary, such as in the function that returned the equivalent distance that corresponded to a given sensor voltage. Instead of using centimeters as a unit, we used millimeters.

5.3 Modular Design Approach

Almost every task that had to be done by the HC11 was implemented in a short, dedicated function. This kept our main program loop neat and short, consisting mostly of one line function calls. In doing so, we were able to write the entire outline and body of the algorithm early on, since debugging a single system required only commenting out the other function calls. Each function was built and tested on its own, then added when needed in its already working state.

Later on in the project, we found that changing the order of various operations was necessary, which was easily done by swapping single lines.

5.4 Defined Constants and Autoscaling

Although changes were anticipated, we had no idea exactly how many changes would be made, especially if we had to backtrack on an entire subsystem. There were too many constants which turned out not to be very constant at all. In fact, the only constant that could be depended on was constant change, from the Analog to Digital converter voltage references to the zeroing of the distance sensor outputs. To save time and temper, all constants used were listed in the very beginning of the program in a list of DEFINE statements. As the robot began running, changes grew more drastic, and it became expedient to add functions that would autoscale all closed loop control K constants with each new change. This was, in essence, closed loop coding. Even at six thirty in the morning, making last minute adjustments were easy and almost idiot proof.

5.5 Mathematical Modeling

In the spectrum of shifting values, there were a few constants that we were able to take advantage of by mathematical modeling. One of them was the response curve of the 2D12 distance sensors. Instead of having a simple ratio of distance to voltage, these sensors produced a curve that was exponential. Rather than using a quantized look-up table of equivalent values or accepting dramatic differences in control at different wall following distances, we modeled the sensor response curve in Matlab. While the curve itself did shift up and down as base voltage varied, the second order relationship remained constant. The resulting equation was used to give us consistent distance measurements over the entire usable range of the sensors (figure 3). This freedom of range was invaluable in our first week of wall following adjustments. In fact, the flexibility of the program in general allowed us to make advances and test new ideas quickly without being heavily lagged by code updates and changes.

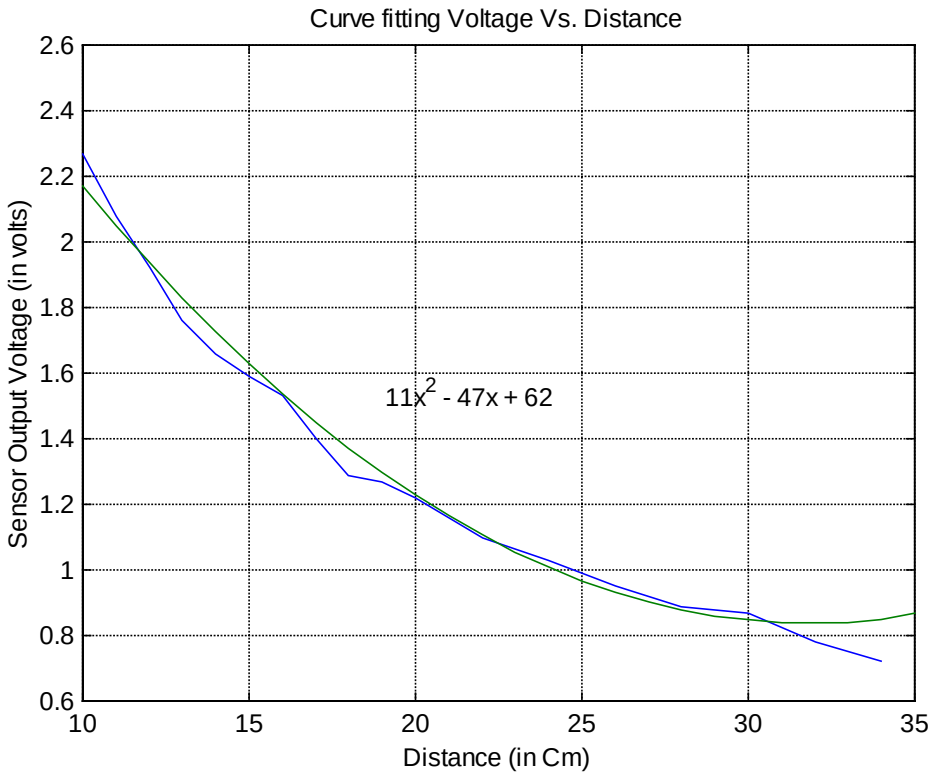
Curve fitting Voltage Vs. Distance
Sensor Output Voltage (in volts)
Distance (in Cm)
11x^2 - 47x + 62
2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
10
15
20
25
30
35

Figure 3

Conclusion

While in the end, troublesome and elusive bugs slowed us down enough to prevent us from completing the robot in time for the competition, we feel we have proven our robot to be a viable design for the task.

Many of the strengths we predicted this design would have did become reality. Power consumption was low with only one motor, slow speed movement was smooth even with big wheels thanks to the gearing down of the differential axle, and wall following was extremely easy to program and implement.

Several drawbacks did repeatedly come back to haunt us, such as the noise sensitivity of the H-Bridges designed for higher voltage radio controlled cars, difficulties in sensor reading fluctuations due to the body swinging from the articulated rear axle, and an underestimate of the time it would take to reverse engineer the RC parts and learn to interface the HC11 and Altera with them.

Of course the entire project was a learning experience; design courses usually are. However even though the course criteria have remained essentially unchanged over the years, the open-ended nature of the class allows infinite possibilities. Even if we had this project to do over again, no amount of time would truly be enough to explore all of the untapped options and design ideas that can be created.

As for the future of Glitchy, we hope to introduce him to all the other competing robots in Connecticut around this time next year.