

# UEzMow2 Autonomous Lawnmower

---

May 15, 2008



University of Evansville

Team

Jeremy Cocco

Kenzie Koehler

Mark Randall

Billy Rickey

## Table of Contents

|  |    |
|--|----|
| Introduction.....                            | 3  |
| 1. High Level Control .....                  | 3  |
| 1.1 UE-Motion Algorithm.....                 | 3  |
| 1.1.2 GPS Coordinate Transformation .....    | 4  |
| 1.1.3 Navigation Through the Waypoints ..... | 5  |
| 1.2 Obstacle Avoiding .....                  | 6  |
| 1.3 Safety Interrupt Routines.....           | 6  |
| 2. Electronics Design.....                   | 7  |
| 2.1 Power Systems.....                       | 7  |
| 2.2 System Design.....                       | 8  |
| 2.3 Crosstalk/Handshaking.....               | 8  |
| 2.4 GPS Implementation.....                  | 9  |
| 2.5 Obstacle Detection .....                 | 9  |
| 2.6 Main Processor .....                     | 10 |
| 2.7 Pre-processor .....                      | 10 |
| 2.8 Motors.....                              | 11 |
| 3. Mechanical Specification.....             | 11 |
| 3.1 Gas Engine .....                         | 11 |
| 4. Cost of Parts .....                       | 12 |
| Conclusion.....                              | 13 |
| Appendix I.....                              | 14 |
| Appendix II.....                             | 15 |

## **Introduction**

The purpose of this project is to design, implement, and test an autonomous lawnmower. The lawnmower should be able to cut a predefined field of grass while avoiding static and dynamic objects. A GPS system will be used to guide our lawnmower and to relay to us where the lawnmower is currently at on the field. Sensors will be used for obstacle detection. If an obstacle is detected, notification should be sent for the robot to either stop before hitting the object or to maneuver around the object. The lawnmower will also include many safety features in hardware and software, such to avoid accidents. The design team for this year consists of Mark Randall as the lead hardware designer, Kenzie Koehler on hardware support, Jeremy Cocco on software development, and Billy Ricky with software support.

### **1. High Level Control**

As the lawnmower maneuvers through the playing field, the high level control algorithm considers a multitude of factors with varying priorities when determining the next action. These factors consist of the robots position with respect to boundaries of the course, the robots desired path, and potential obstacle blocking this path. As mentioned before, some factors have priority over others. For example, it is much more important to avoid running someone or something over than maintaining a straight path. Considering these priorities, the main processing loop contains a hierarchy of control algorithms that need to be followed.

#### **1.1 The UE-Motion Algorithm**

The UE-Motion algorithm is the main algorithm used by the main processor. Although it will and can be interrupted by things such as GPS, sensor data transmission, and emergency stop mechanisms, it is the brains behind how our lawnmower moves throughout the field. Based on the velocity of our lawnmower, and the calculation of waypoints prior to the competition, we can find the best possible route for our lawnmower to follow. The lawnmower will follow a series of waypoints and cut the grass as fast as it can and then move onward to the next waypoint.

### **1.1.2 GPS Coordinates to Cartesian Coordinates**

One of the most essential parts of our lawnmower is the GPS system and using it to build a Cartesian coordinate field. The Cartesian coordinate field simplifies the lawnmower control algorithm because it is an easy way to interpret the playing field by simply looking at the field as a series of x and y direction waypoints. These waypoints describe the positions that the lawnmower will have to encounter while cutting grass. Waypoints can then be connected such as to form a path during the lawnmower's run. A MATLAB program is then able to calculate a path considering the size of the robot, linear velocity, and turning velocity. The program can break the path into a series of waypoints, which are loaded into the robot.

The BD950 GPS system we will be using relays information serially (RS232) in the form of GPGGA NMEA sentences. The global latitudinal and longitudinal information is in the form of degree-minutes. An example output for latitude would be 3758.27032425 which can be interpreted as 37 degrees and 58.27032425 minutes. In terms of degrees and minutes, this number is above and beyond what we need to mow a field. The reason being is that 1 degree for latitude counts as approximately 69 miles and 1 degree for longitude counts as approximately 42 miles. This shows that a good portion of these values are unnecessary because they would reflect miles and miles of travel, which our lawnmower can't obviously do. This is why we toss out some of the digits and make use of only a few of them at the end of the latitude and longitude outputs, because they describe the small changes made when our lawnmower moves.

After recording the positions of all the corner points in our playing field (6 total), we can now plot a Cartesian coordinate field. A MATLAB program we've created is a function that takes six inputs as arguments. Each argument will be a vector that describes the corner point of our playing field. We can then normalize the field such that our first corner point is the origin of our Cartesian coordinate playing field. Then we apply to the points to a MATLAB built rotate function that will rotate the points such that we get a field right-side up. Then we also make sure our original point is at the origin and thus we translate all the points such as to keep the shape of the field. This simulation will reflect what needs to be done in software. Now we have a series of points in the Cartesian coordinate system that we can use to calculate how many waypoints will be needed. A simulation for this is provided below in figures 1 and 2.

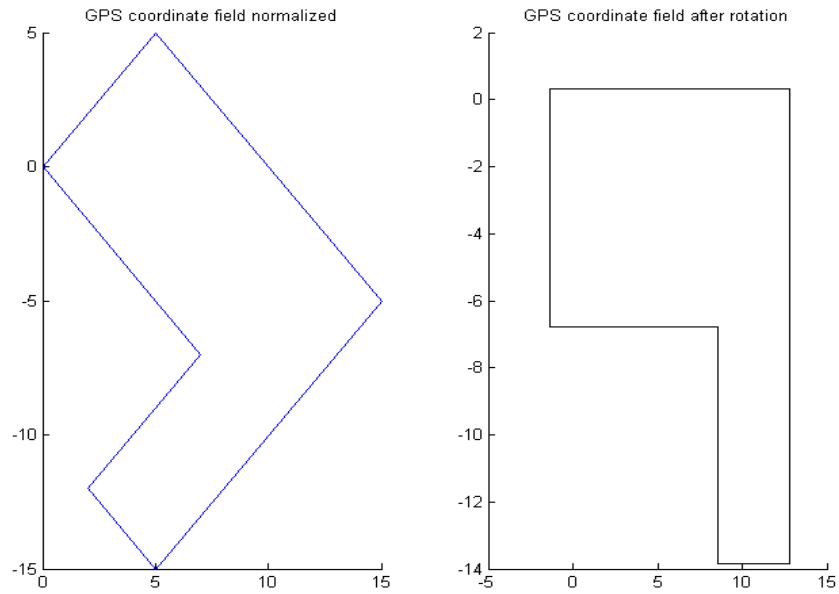


Figure 1: Example coordinate

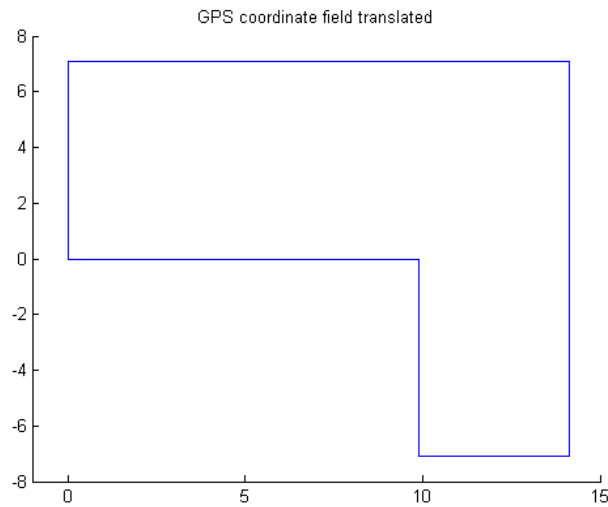


Figure 2: Example of coordinate

### 1.1.3 Navigation Through the Waypoints

As mentioned earlier, once we are able to map the GPS coordinate system to a Cartesian coordinate system, we would be able generate a series of waypoints that would map the places the lawnmower needs to mow. It's up to us to determine how we want to navigate through the

field. Since we have waypoints in terms of the x and y position, we can take linear paths that either make x-position movements or that take y-position movements at one time. Using this method, we can easily track where the lawnmower has been and which waypoint it needs to follow next. We will also make use of waypoints to make sure that the lawnmower is on track. If the lawnmower was to veer off course, we will compensate for the wheel motor that is either falling behind or leading the other one in turns. We want to make sure that the lawnmower is going at the desired speed we chose it to be. Therefore, since we know where waypoints are, and when we are supposed to reach them, we can control how the lawnmower should navigate through the course with relative ease.

### **1.2 Obstacle Avoiding**

There are two obstacles that we must avoid during the operation of lawnmower. The first is the dynamic object (the fun and lovable plush Poodle on wheels). We are using three sonar rangefinders to measure the distance away from the object. Our program on the pre-processor constantly finds the distance from an object and these distances get interpreted by the main processor. We also send information on which sensor it was that sent the data. The main processor then decides if the lawnmower is in danger by checking the current distance against a fixed distance. If the lawnmower is not in danger, then it will continue on its path through the waypoints. Otherwise, the lawnmower will stop, before it hits the moving object, and wait for it to move.

The flowerbed is a different story. We will be using four IR distance sensors to locate the flowerbed and to navigate through the semi-circular area. The sensor output voltages will then be put through an ADC and an 8-bit word will be passed to an 8-bit address bus for each additional sensor. This will then give us an idea of how close we are to an edge with a particular sensor and, from there, adjustments would be made to the speed control algorithm such as to navigate through the semi-circular area.

### **1.3 Safety Interrupt Routines**

This year we must, yet again, implement safety features that keep the lawnmower in check and, also, to prevent us from losing any points during the competition. First off, if the robot ends up running out of the safety buffer zone, we must cut power to both the blade and the wheelchair motors. This will be done as a function in software. We will also setup the routine such that if the lawnmower veers into the safety buffer because it needs to make a turn, we must

cut power to the blade to prevent cutting grass in the safety buffer zone and write a routine to make it turn back on course.

We will be using a wireless key fob to shut off the lawnmower remotely. A UHF Remote responds to the signal sent by one of us. The response for the processor is then to almost immediately shut down the motors of the lawnmower. We have also included a manual stop button that works much the same way as the key fob does, but kills all power to the entire robot rendering it useless.

## **2. Electronics Design**

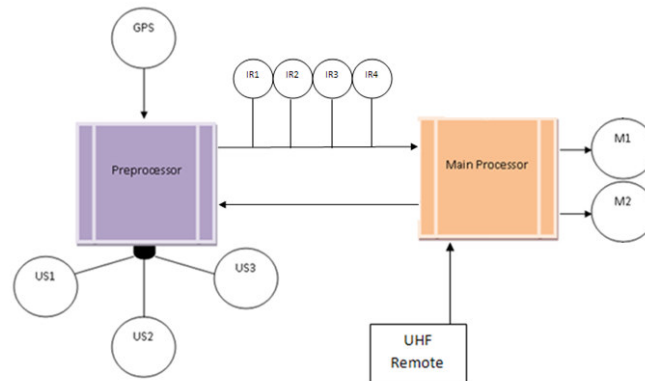
The electronics design of this project focused in on a couple tasks that were necessary for getting the lawnmower up to speed. First, we must be able to get the GPS data from the GPS receiver, be able to parse the GPS data to useable data for our pre-processor board, and build a virtual playing field from this data. Second, we need to avoid all obstacles that are in the path of the lawnmower. We must interpret the distances from the lawnmower to the object, evaluate them and see if we need to alter or halt the path of the lawnmower, and be able to revert back to the lawnmowers normal path after we have passed the object. The system we built for this uses two ATMEL 8051 processors: The 89C51ED2 (our main processor) and 89C51RD2 (our pre-processor).

### **2.1 Power Systems**

For power, our mower will consist of both a gas and a 12 volt power source. The gas power source will mainly be used to move the blade, considering it would take far too much amperage for the electric part of the power system. The 12 volt source is used to power both motors and is regulated down to two five volt isolated power supplies that will be used to power both of our CPU boards (this includes processor chips, sensors, etc.) and the GPS. This is done using two TDK-Lambda CC6-1205 isolation chips. The isolated five volts is important, because it removes motor noise from the logic power/ground busses. This kind of noise could easily corrupt analog data and possibly cause the processors to reset. Our main board uses a magnetic isolation chip (ADuM1400) to complete the separation between our logic outputs and the h-drivers. Also, the h-driver we're using, as described later, provides a 5 volt output to be used by a R/C Futaba receiver and by the isolator we just mentioned.

## **2.2 System Design**

The system that we've created is built around two Atmel 89C51 processors, with the main processor being an Atmel AT89C51ED2. This processor was selected because of the processor's unique features, as well as our vast knowledge of 8051 architecture. The main processor will be using data collected from the preprocessed GPS data, the obstacle data to avoid contact with the dynamic and static objects, and other data from our safety sources to make decisions for the lawnmower. The second processor is the Atmel AT89C51RD2. Its purpose is to preprocess any data received from the GPS systems as well as the sensors. This processor will translate the raw data from the two devices to make it easier for the main controller to use, thus reducing the processing time a great deal. A simple example of our system can be found in figure 3 below.



**Figure 3: Simple system Diagram of multiprocessor configuration.**

## **2.3 Crosstalk/Hand Shaking**

The handshaking process is just like last year's lawnmower. The pre-processor (89C51RD2) needs to communicate with main processor (89C51ED2) by sending information over an 8 bit address bus. This will be information from the sensors and the GPS. The main processor will make a request for information from the pre-processor. This request goes to a tri-state transparent latch (74HCT373) that stores the request and an external interrupt is triggered by the pre-processor. The pre-processor sets itself to read from the latch and writes the information requested to another tri-state transparent latch. In doing so, the main processor triggers its external interrupt and sets itself to read the requested information. Thus the

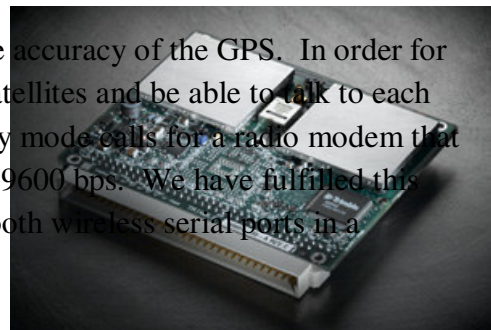
handshaking between the processors is possible. Hence, this method is a good way to send a receive data without crashing the bus.

## **2.4 GPS Implementation**

Just like our last year's autonomous lawnmower, we have decided to use the Trimble BD950 GPS receiver. The accuracy of the unit itself and the cost for which we bought it made it ideal for our GPS needs. As we mentioned before, the BD950 is accurate up to 2cm and supports three modes: standalone, differential, or Real-Time Kinematic (RTK). We have decided, like last year, that to get the accuracy necessary for this project we would need to program the

GPS unit in RTK mode. RTK mode requires a GPS unit on the machine and a "Base Station" from which distance and phase calculations can be done to increase the accuracy of the GPS. In order for this to work, the two GPS units must have five common satellites and be able to talk to each other. The specification for the BD950 in RTC low latency mode calls for a radio modem that can transmit standard RS232 data at a baud rate of at least 9600 bps. We have fulfilled this requirement by using two "BlueSMIRF" 100 meter Bluetooth wireless serial ports in a slave/master configuration.

**Figure 4: Picture of BD950 GPS Receiver**



## **2.5 Obstacle Detection**

Because of the ease and performance of last year's ultrasonic rangefinders, we decided to use the Maxbotix LV-EZ1 rangefinders for dynamic obstacle detection. The resolution of the sensor is more than enough for detecting objects and they are well suited for the processors we are using. The LV-EZ1 can be interfaced with a microprocessor using pulse width, analog voltage, or serial digital output. As with last year's lawnmower, we decided to use pulse width output because the 89C51RD2 processor, that we will be reading the pulse width from, uses the Programmable Counter Array (PCA) to capture the rising and falling edge of a signal. The signal's high time can be measured and interpreted into inches of travel, where 1 inch is the equivalent of 147uS high time. To fire a



**Figure 5: Picture of Maxbotix LV-EZ1**

signal, a logic 1 must be sent to the RX pin of the rangefinder. Our final implementation will make use of three rangefinders. The sensors will be mounted at 45 degree angles from one another. These sensors will give a 90 degree span total of the playing field, which should be more than plenty to find the location of the dynamic object.

In addition to our sonar rangefinders, we also used distance IR sensors to locate the flowerbed. We've decided, based on cost and precision, that the Sharp GP2D120 IR sensors were the best choice for this project. These sensors can detect objects from a minimum distance of 4 cm to a maximum distance of 30 cm.

## **2.6 Main Processor**

The monitoring of the wireless kill (emergency) switch is a job left up to the main processor. The kill switch is a logic device that is simply connected to a certain set of I/O pins. The main processor is also responsible for the controlling of the motors. The motors are attached to the last three modules of the PCA, which are setup in pulse width modulation (PWM) mode.

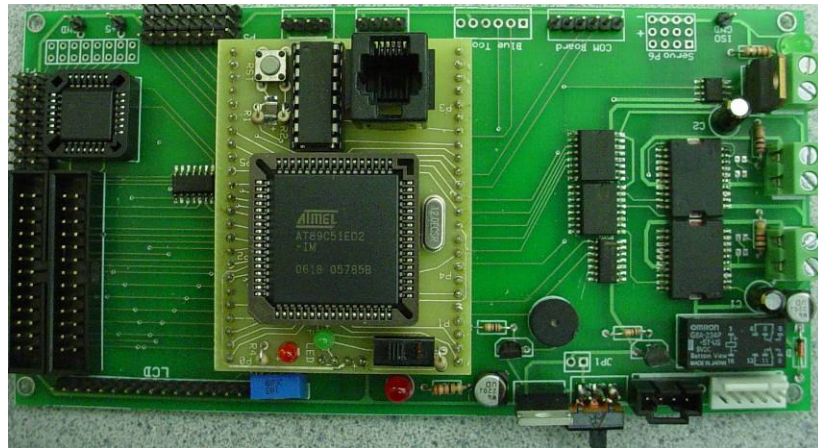


Figure 6: Picture of AT89C51ED2 main robot controller PCB.

Schematic found in appendix I.

## **2.7 Pre-processor**

The second processor that we've used will preprocess data received from the GPS and ultrasonic sensors. The GPS is connect to the processor via a standard RS232 port and standard ASCII characters are transmitted to the processor in a comma separated fashion. The processor downloads the data, parses out what is needed to create an 8 bit number that specifies the robot's location, and downloads that parsed data to the main processor. It then does all the calculations necessary to translate the data from the global coordinate system to a relative coordinate system that the main processor can use. This processor will also take

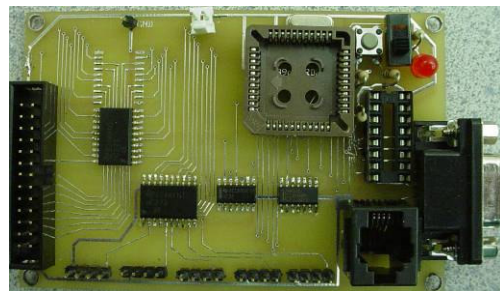


Figure 7: Picture of AT89C51RD2 Pre-processor PCB . Schematic found in appendix II

care of pre-processing the data from the ultrasonic sensors. The three ultrasonic sensors will be connected to the processor using three of the PCA channels. Also, additional I/O was needed to fire each of the sensors separately. To correct this problem, an output port was added to the I/O bus which can now be used to start each sensor. The processor will be able to translate the pulse width measurement received to inches and download this data to the main processor.

## **2.8 Motors**

The drive motors that we're using are two, high-torque, wheelchair motors that came with our lawnmower base. They are both 12 volt DC with a 200W power rating. The two drive motors have a gear ratio of 32 to 1 and go at a speed of around 110 RPM. Also, as stated above, the motors are being driven with the microprocessor by a PWM signal. We've decided to use a Sabertooth dual 25 amp motor driver for this purpose. The driver outputs a 32kHz modulated PWM output signal. This driver can also be put into multiple modes, such as analog voltage, radio control, serial, and packetized serial. The obvious mode for us to use is the radio control input mode.

## **3. Mechanical Specification**

The robot has two 12 volt DC motors for driving the wheels and one gas motor for the blade. The outside dimensions of the mower consisted of a length of 42 inches, a width of 24 inches, and a height of 30 inches. The mower is equipped with one cutting blade that has a span of 21 inches. The drive motors drive two, eight inch diameter and two inches thick, front tires and two, 10 inch diameter and three and a half inch thick, knobby back tires. For power we're using both gas and a 12 volt rechargeable battery along with a 60A alternator which will help to charge the battery while the mower is on, thus conserving a little power. On average our mower had a cutting time of around 1-1.5 hrs per charge/fill-up.

### **3.1 Gas Engine**

The engine is a 6.75 hp Craftsman electric start push mower engine. It is outfitted with a 12 volt DC starter motor which is switched on and off by the main CPU via a 12V SPDT relay. In addition to the "Start Relay" an additional DPDT relay is being used for safety purposes. This DPDT relay in the normally off state has the main coil on the engine grounded, and "started relay" out of circuit. The relay is driven using reverse logic so a logical 0 turns on the machine

relay. As is true with most microcontroller all of the port pins on an AT89C51 processor come up in the “High” state or a logical 1. This allows for a failsafe condition that will not allow the engine to run or start if the microcontroller is functioning incorrectly.

#### **4. Cost of Parts**

Here is an approximate cost breakdown for this year’s lawnmower. As you can see the bulk of the cost rested upon the GPS receiver and antennas along with the lawnmower body, motors, and wheels.

##### UEzMow2 Cost Breakdown

| <u>Item</u>                          | <u>Quantity</u> | <u>Cost</u> |
|--------------------------------------|-----------------|-------------|
| Lawnmower Body, Motors, Wheels       | 1               | \$1300.00   |
| Range Finders <u>Maxbotix</u> LV-EZ1 | 3               | \$74.85     |
| IR Sensors (Sharp GP2D120)           | 4               | \$50.00     |
| 12V Deep Cycle battery               | 2               | \$130.00    |
| Board Layouts and Production         | 1               | \$292.00    |
| Board Population                     | 1               | \$200.00    |
| Manual Emergency Stop Mechanism      | 1               | \$37.85     |
| Bluetooth Modem                      | 2               | \$120.00    |
| GPS BD950 w/ Antennas                | 2               | \$4000.00   |
| 4-CHANNEL UHF REMOTE CONTROL         | 1               | \$30.00     |
| <u>Sabertooth</u> (2x25 controller)  | 1               | \$128.00    |
| Craftsman 6.75HP Gas Engine          | 1               | \$300.00    |
| Misc Parts (approx)                  | 1               | \$500.00    |
| Futaba R/C controller                | 1               | \$200.00    |

After totaling up the individual costs, our mower was a bit pricey summing around \$7462, but considering our results so far, we think it was worth it.

## **Conclusion**

After working for many months on getting our lawnmower up and running, we believe that we've got something worth entering in this competition. We've tested both hardware and software and are finally getting to the point where we can put everything together to make it work. We are anxious to get the mower completed and hope to see it cut grass very soon. We hope that the rest of the time spent finishing up the lawnmower will be very productive and that we will end up with another successful robot that our team and the University of Evansville will be proud to stand behind.

## **Special Thanks**

We'd also like to thank Addisu Taddese and Zachariah Fuchs for their previous documentation and research on this project.



