GPS Guided Robotic Car Progress Report

Prepared for: Dr Jason Blough
SURF

Prepared by: Alex Morozov
May 29, 2008

“GPS Guided Robotic Car Progress Report”

Objective

This document will be covering all the achievements made by the student while working on this research project.

Discussion

In the beginning of the summer 2008 semester GPS-test units as well as IMU, microprocessor, LCD and break-out boards were acquired (for datasheets, please refer to appendices). The completion of the project from that point on involves writing the software to control the communication flow between the components of the system. There were a couple of decisions to be made, such as what programming language to use and how many GPS units to put on the car. As of right now the language used is assembly due to its ability to manipulate any kind of information on a very low level (high level of control). The amount of GPS units to be put on the car will possibly change in the future. Right now there is one module attached to the car and all it does is selectively save NMEA lines onto an SD card. Even though this device might seem primitive and not entirely connected to the main goal of the project, it provides invaluable experience of parsing NMEA sentences and decoding actual geophysical data out of these sentences.

The next thing to do would be to connect the Inertial Measurement Unit to the GPS and to the Microcontroller and write a Kalman Filter to combine these two sources of information to provide more precise position, velocity and acceleration numbers.

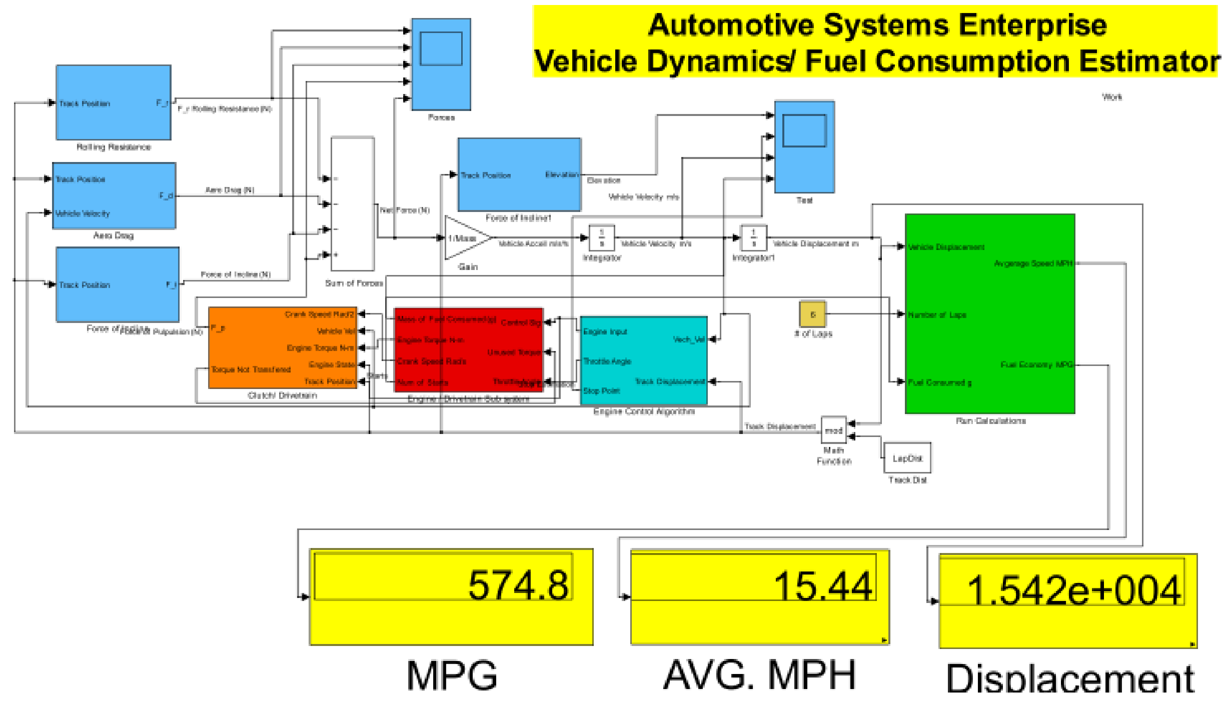
Further plans include installing the finished system on a test vehicle (an RC-car) to demonstrate and double-check the functionality and accuracy of the system.

In the background of this research, the Supermileage vehicle is being completed. Among the latest achievements are:

- Finishing the body-shell (see appendicies)

- Testing the EFI system
- Upgrading the MATLAB Simulink Model (see appendicies)
- Verifying the energy budgets
- And multiple minor achievements like troubleshooting and redesigning failed parts of the system

Appendix 1. Supermileage model

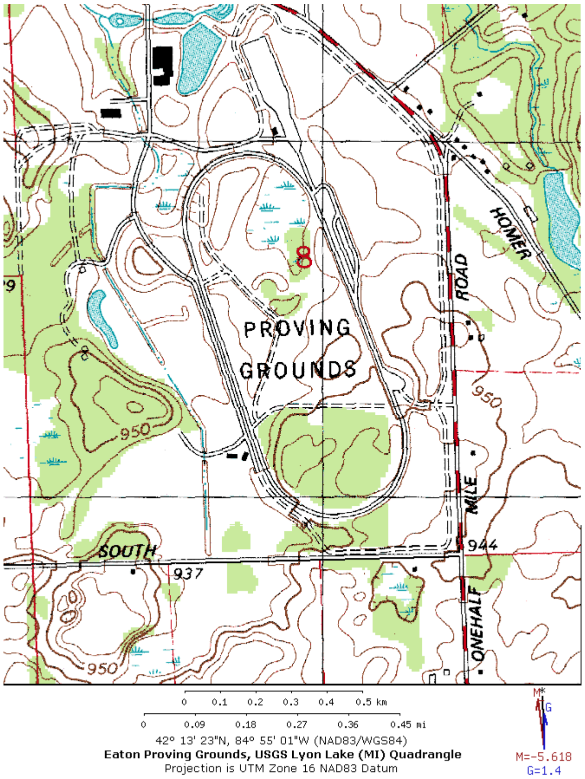


Please go to www.ece.mtu.edu/~amorozov/robo/ for a full model description.

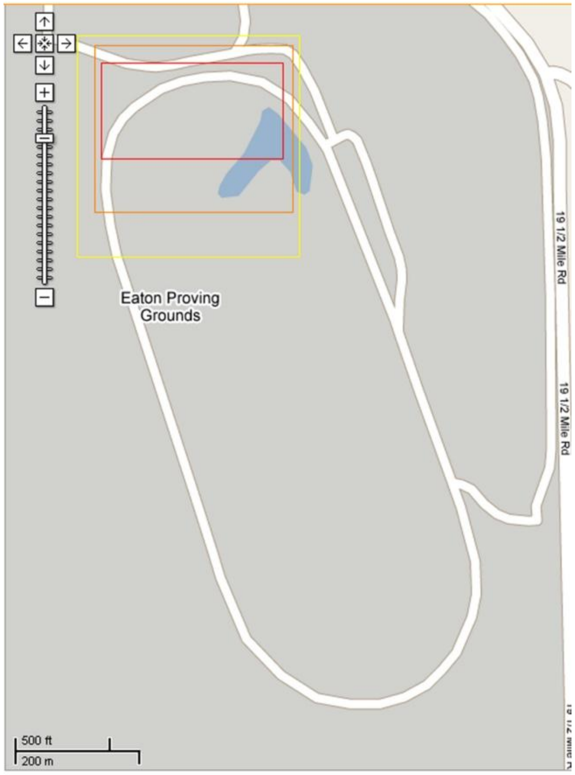
Appendix 1. Supermileage car

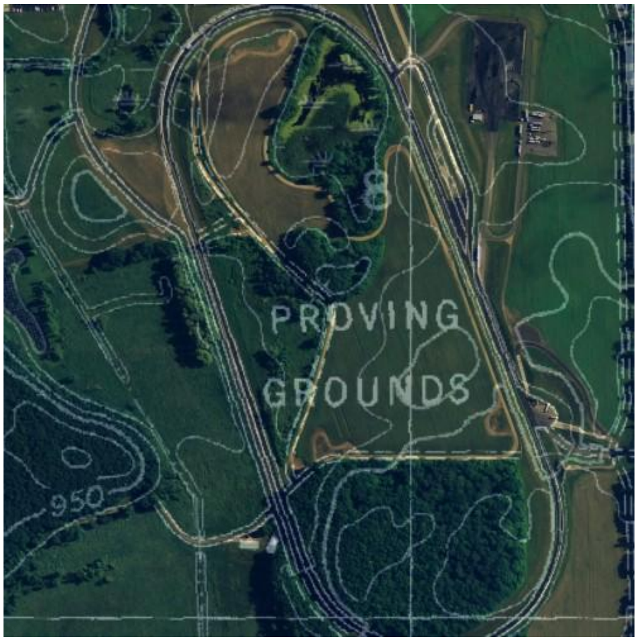
For more information about the Supermileage car in particular and Automotive Systems Enterprise in general, please go to www.enterprise.mtu.edu/ase

Appendix 2. Maps



42° 13' 23"N, 84° 55' 01"W (NAD83/WGS84)
Eaton Proving Grounds, USGS Lyon Lake (MI) Quadrangle
Projection is UTM Zone 16 NAD83 Datum

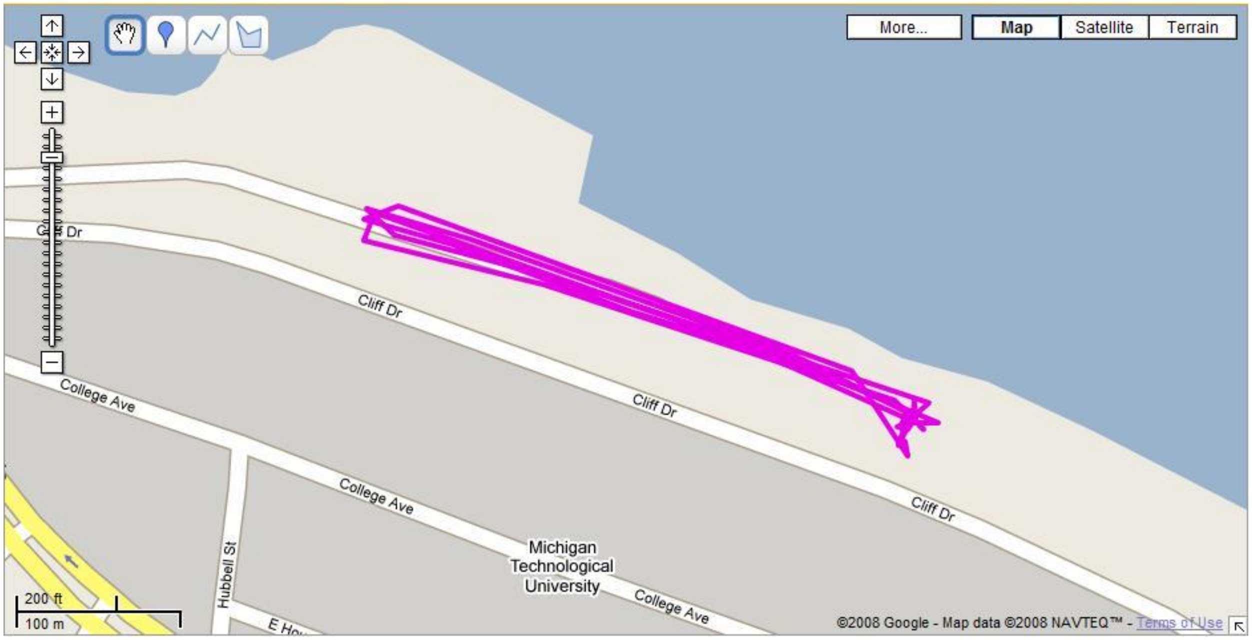




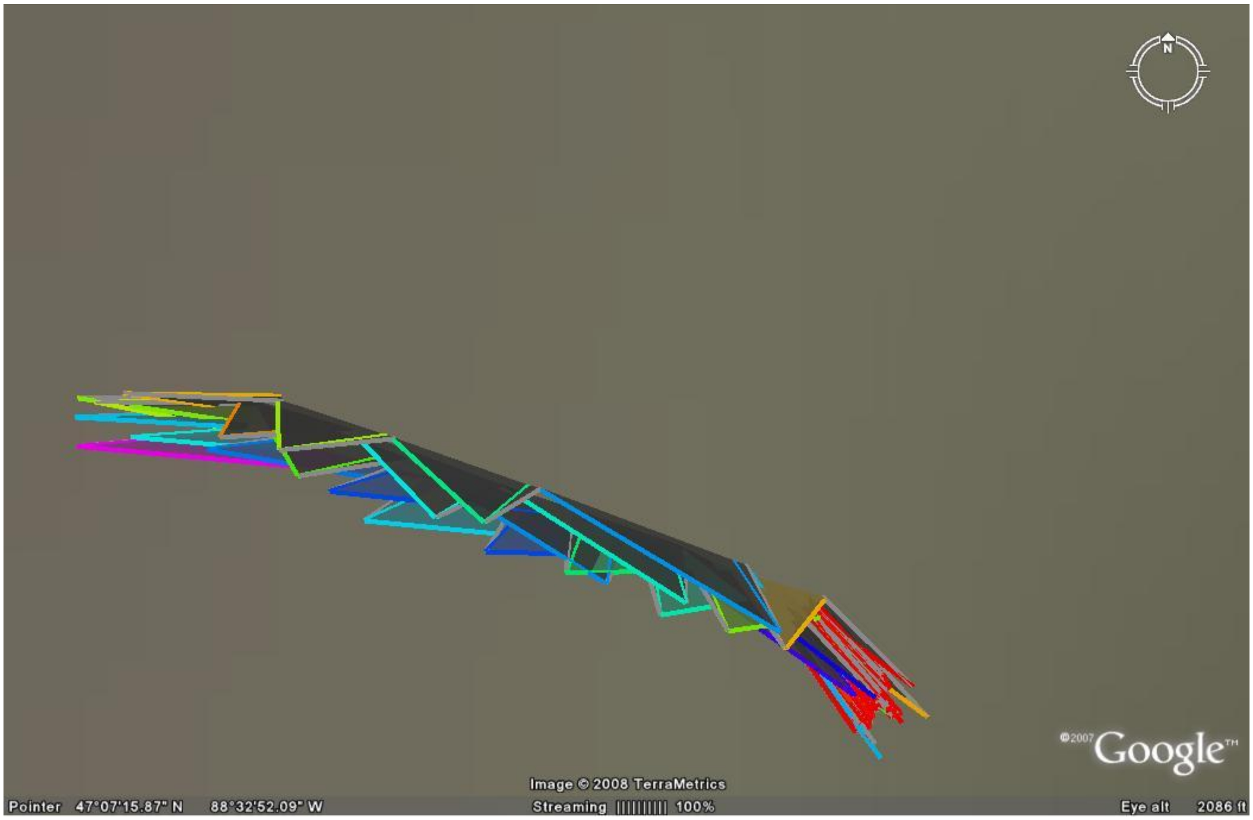
Appendix 3. Sample NMEA 0183 data

```
$GPRMC,204910.000,A,4707.2458,N,08832.9333,W,3.21,143.23,290508,,*1A
$GPGGA,204911.000,4707.2448,N,08832.9326,W,1,05,3.0,191.3,M,-35.0,M,,0000*67
$GPGSA,A,3,31,14,20,29,23,,,,,,,,3.5,3.0,1.7*3E
$GPGSV,3,1,12,25,82,014,,01,80,356,,31,73,047,41,20,47,287,41*70
$GPGSV,3,2,12,16,31,175,28,14,26,106,37,30,23,046,31,23,21,295,38*70
$GPGSV,3,3,12,05,13,041,20,06,10,076,,29,09,078,36,07,04,122,*76
$GPRMC,204925.000,A,4707.2387,N,08832.8953,W,13.67,100.76,290508,,*20
$GPGGA,204926.000,4707.2377,N,08832.8900,W,1,05,3.0,189.7,M,-35.0,M,,0000*6A
$GPGSA,A,3,31,14,20,29,23,,,,,,,,3.5,3.0,1.8*31
$GPGSV,3,1,12,25,82,014,,01,80,356,,31,73,047,41,20,47,287,40*71
$GPGSV,3,2,12,16,31,175,33,14,26,106,35,30,23,046,43,23,21,295,38*7D
$GPGSV,3,3,12,05,13,041,23,06,10,076,,29,09,078,35,07,04,122,*76
$GPRMC,204940.000,A,4707.2161,N,08832.8046,W,15.03,109.68,290508,,*26
$GPGGA,204941.000,4707.2148,N,08832.7988,W,1,05,3.0,189.4,M,-35.0,M,,0000*69
$GPGSA,A,3,31,14,20,29,23,,,,,,,,3.4,3.0,1.8*30
$GPGSV,3,1,12,25,82,014,,01,80,356,,31,73,047,43,20,47,287,40*73
$GPGSV,3,2,12,16,31,175,21,14,26,106,37,30,23,046,44,23,21,295,40*74
$GPGSV,3,3,12,05,13,041,32,06,10,076,,29,09,078,41,07,04,122,*75
$GPRMC,204955.000,A,4707.1992,N,08832.7261,W,15.82,106.13,290508,,*27
$GPGGA,204956.000,4707.1976,N,08832.7197,W,1,05,2.9,193.1,M,-35.0,M,,0000*69
$GPGSA,A,3,31,14,20,29,23,,,,,,,,3.4,2.9,1.8*38
$GPGSV,3,1,12,25,82,019,,01,80,001,,31,72,048,42,20,48,286,39*70
$GPGSV,3,2,12,16,32,175,20,14,25,107,37,30,23,046,44,23,21,296,40*77
$GPGSV,3,3,12,05,12,040,37,06,11,075,,29,09,077,40,07,04,122,*7C
$GPRMC,205010.000,A,4707.1804,N,08832.6557,W,6.38,157.28,290508,,*1C
$GPGGA,205011.000,4707.1788,N,08832.6551,W,1,05,2.9,194.4,M,-35.0,M,,0000*60
$GPGSA,A,3,31,14,20,29,23,,,,,,,,3.4,2.9,1.8*38
$GPGSV,3,1,12,25,82,019,,01,80,001,,31,72,048,41,20,48,286,42*7F
$GPGSV,3,2,12,16,32,175,,14,25,107,39,30,23,046,39,23,21,296,40*71
$GPGSV,3,3,12,05,12,040,33,06,11,075,,29,09,077,40,07,04,122,*78
```

Appendix 3a. Sample NMEA 0183 data plotted on the map



A test run down by the M&M/Dow buildings


N
©2007 Google™
Image © 2008 TerraMetrics
Pointer 47°07'15.87" N 88°32'52.09" W
Streaming ||||||||| 100%
Eye alt 2086 ft

The same run, but colorcoded by velocity – bright red means stationary, dark blue means maximum velocity (25 mph).

Appendix 4. Sample assembly code used to test the microprocessor

```
                              ;Binary Counter
                              ;Alexey Morozov
                              ;
                              ;This program makes the AVR 128 output binary numbers
                              ;in decreasing fashion and hence "count"
                              ;8-LED array with resistors should be connected to
                              ;Port C



.include "m128def.inc"  ;Includes the 8515 definitions file



.def OUTPUT = R16       ;Names Register R16 OUTPUT
.def count = R17        ;Names Register R17 count
.def count1 = R18       ;Names Register R18 count1
.org 0x0000             ;Places the following code from address 0x0000
  rjmp  RESET           ;Take a Relative Jump to the RESET Label

RESET:

  ldi OUTPUT, 0xFF      ;Store 255 in R16 (start counting from 255 down)
  ldi count, 0xFF       ;Store 255 in R17 (start counting from 255 down)
  ldi count1, 0x0A      ;Slow down-factor of ten
  out DDRC, 0xFF         ;Make PORTC an output port

Loop:

  out PORTC, OUTPUT  ;Write all highs (255 decimal) to PORTC
  dec OUTPUT            ;Decrement the register

Firstloop:

  nop                   ;Slow down the execution by applu nop (no action)
  dec count             ;Decrement the loop counter
  brne Firstloop ;If the cycle was completed, loop back and start over
  nop                   ;Wait
  dec count1            ;Decrement the other loop's counter
  brne Firstloop ;If the outer cycle was completed too, restart

  rjmp Loop             ;If nothing was completed, just continue looping
```