IOWA STATE UNIVERSITY

EMERGENCY TRACTOR BEACON

A co-operative project between the Electrical and Computer Engineering Department and the Department of Industrial Design

John Pritchard, David Ringholz, Mani Mina, Jamie Horwitz, Michael Tschampl, Mitchell Hinrichsen

7/9/2012

The purpose of this document is to keep a working record of the motivation, design, specification, and changes of the Emergency Tractor Beacon project.

CONTENTS

Introduction 2
Solution Outlook 2
Detailed Solutions 2
Prototype 3
Systems Level Design 3
Electronics Design 4
Software Design 4
Mechanical Description 7
Specifications 7
Level of Effort 7
Cost Analysis 7
Appendix 8
Prototype Software 8
Arduino Code 8
MATLAB Code 9
Index ..... **Error! Bookmark not defined.**

Figure 1: Prototyped wireless transceivers 3
Figure 2: Prototype systems-level diagram 4
Figure 3: Prototype electronics diagram 4
Figure 4: Prototype Arduino software flow chart 5
Figure 5: Prototype MATLAB software flow chart 6

Table 1: Brief outlook of proposed solutions 2
Table 2: Prototype cost analysis 7

INTRODUCTION

It is a continued problem that people are fatally injured due to tipping or even rolling tractors built after the early 1970's. No general solution has been proposed to help aid in the event of such an occurrence, until now. This document proposes three solutions, each with their respective advantages and disadvantages, which signal local authorities in the event of a tip or roll to quickly respond. In addition, a working prototype is described and presented, showing the proof of concept.

The structure of this document is designed to provide a quick overview of the proposed solutions, so the reader can choose which is appropriate for their application. Each solution is then discussed in detail, presenting system descriptions, components, requirements, specifications, level-of-effort, and cost analysis.

SOLUTION OUTLOOK

This section briefly discusses and compares the proposed solutions, highlighting the most important design tradeoffs.

Specification	*Prototype*	*Basic*	*Intermediate*	*Advanced*
Power Requirement	Low	Low	Medium	High
Com Distance*	200m	>3mi.	Global	Global
Usability	Easy	Easy	Moderate	Cumbersome
Sys Complexity	Simple	Simple	Moderate	Complicated
Maintenance	N/A	Low	Moderate	Very High
Level-of-Effort	~2 wks.	~1-2 mo.	~ 3-4 mo.	8-12 mo. minimum
Reliability	N/A	Moderate	High	High
Multidisciplinary Impact	N/A	Low	Moderate	Very High
Cost	~$250	~$400	~$800	>$4000

*Com Distance refers to how far the wireless transmission communicates. Upon successful wireless transmission, authorities will be notified over Ethernet (the internet).

Table 1: Brief outlook of proposed solutions

DETAILED SOLUTIONS

This section describes each proposed solution in detail. The following are provided for each solution:

- Systems-level design
- Electronics design
- Software design
- Mechanical descriptions
- Specifications
- Level-of-effort (LOE)
- Cost analysis

The systems-level design describes how all main components are connected, and how and when information is transferred. It is the description of how the system works at the highest level.

The electronics design describes the hardware involved, generally from a high level perspective. Low level descriptions will be provided when custom circuitry is designed.

The software design describes the software that will need to be written to accomplish tasks given by the specifications and requirements.

The mechanical descriptions provide all physical characteristics of components used including dimension, material, and durability.

The specification description provides a list of capabilities of the solution at hand.

The LOE descriptions provides how much time and resources are required to realize the solution

The cost analysis provides a list of all components, how to purchase them, how much they cost, and alternatives for cost reduction of applicable.

PROTOTYPE

The prototype is a small-scale wireless system (Figure 1) in which the transmitter transmits a warning if the angle of the tractor (with respect to the horizon) reaches beyond a defined threshold.

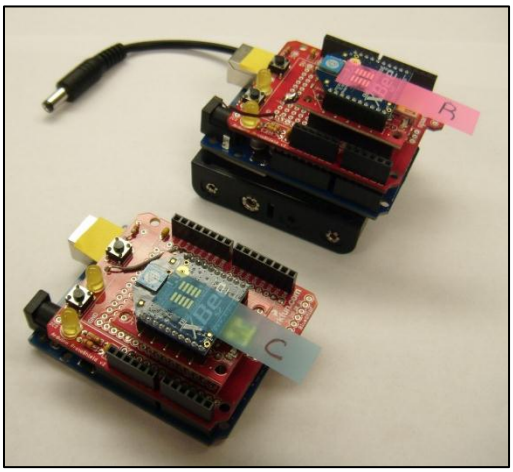

Figure 1: Prototyped wireless transceivers

SYSTEMS LEVEL DESIGN

In the prototype, one transceiver is connected to a PC and awaits the wireless transmission of the warning signal. The other transceiver is attached to the tractor and monitors the tractor's orientation relative to the horizon. If the tractor angles beyond a defined threshold, the attached transceiver will immediately transmit a warning, which will be interpreted by a PC computer on the other end (see Figure 2)

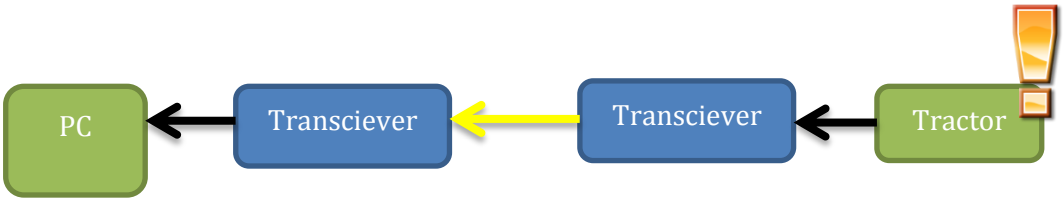

Figure 2: Prototype systems-level diagram

Electronics Design

In this section, each transceiver is defined.

The transceiver attached to the PC contains the following:

- Arduino (electronic controller)
- Xbee (transceiver module)

The transceiver attached to the tractor contains the following:

- Arduino (electronic controller)
- Xbee (transceiver module)
- Electronic Gyroscope (gyro)

The Arduino is a common electronics platform that allows for fast prototyping and interfacing of other electronics sensors, communication devices, etc. The Arduino can handle the information provided by the gyro and the Xbee.

The gyro measures the angular velocity of a given axis, and outputs a voltage that is representative of this rotation. The Arduino converts this voltage to a digital value, and manipulates the data to extract degree information.

The Xbee is an advanced wireless device that can easily interface with the Arduino controller. The range on the particular devices used is near 200 meters line-of-sight.

A connection diagram of this system is provided:

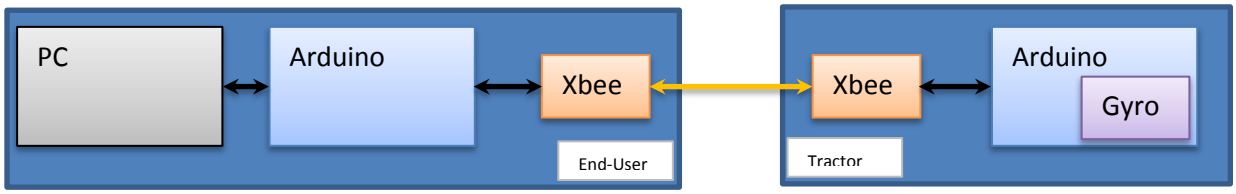

Figure 3: Prototype electronics diagram

Software Design

This section discusses the design of the software. It is split into two sections, since two programming environments were used: Arduino and MATLAB. The Arduino software design

controls the sensor and wireless information handling. The MATLAB software design controls external notification (Google maps and cellular notification).

Arduino Software

The flow chart describing this software can be found below. The code is given in the appendix.

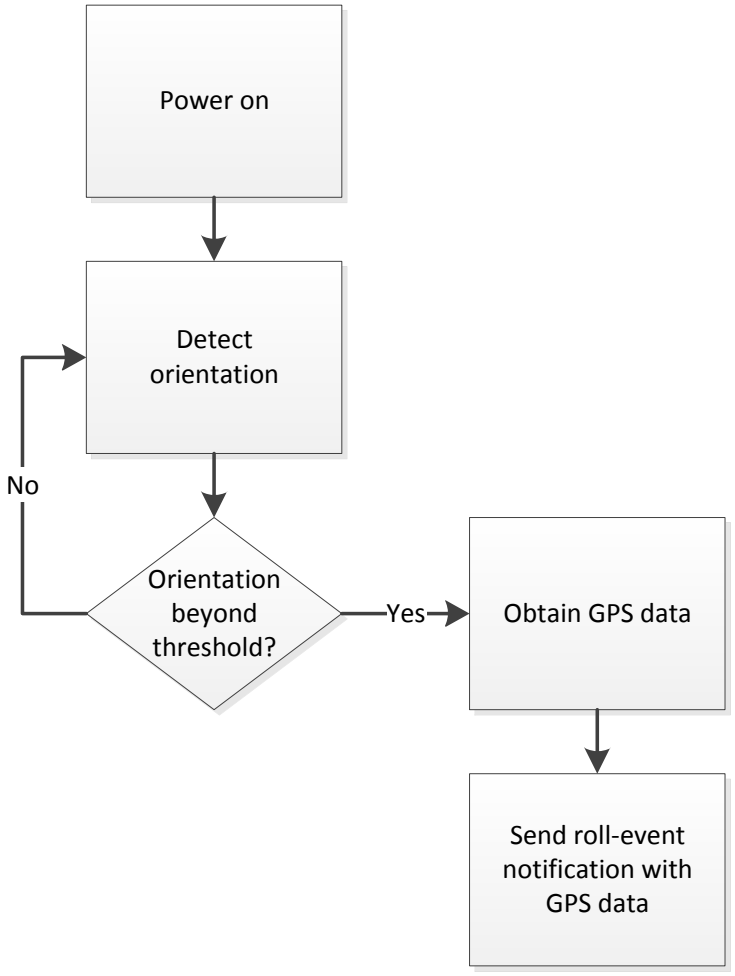

Figure 4: Prototype Arduino software flow chart

MATLAB Software

The flow chart describing this software can be found below. The code is given in the appendix.

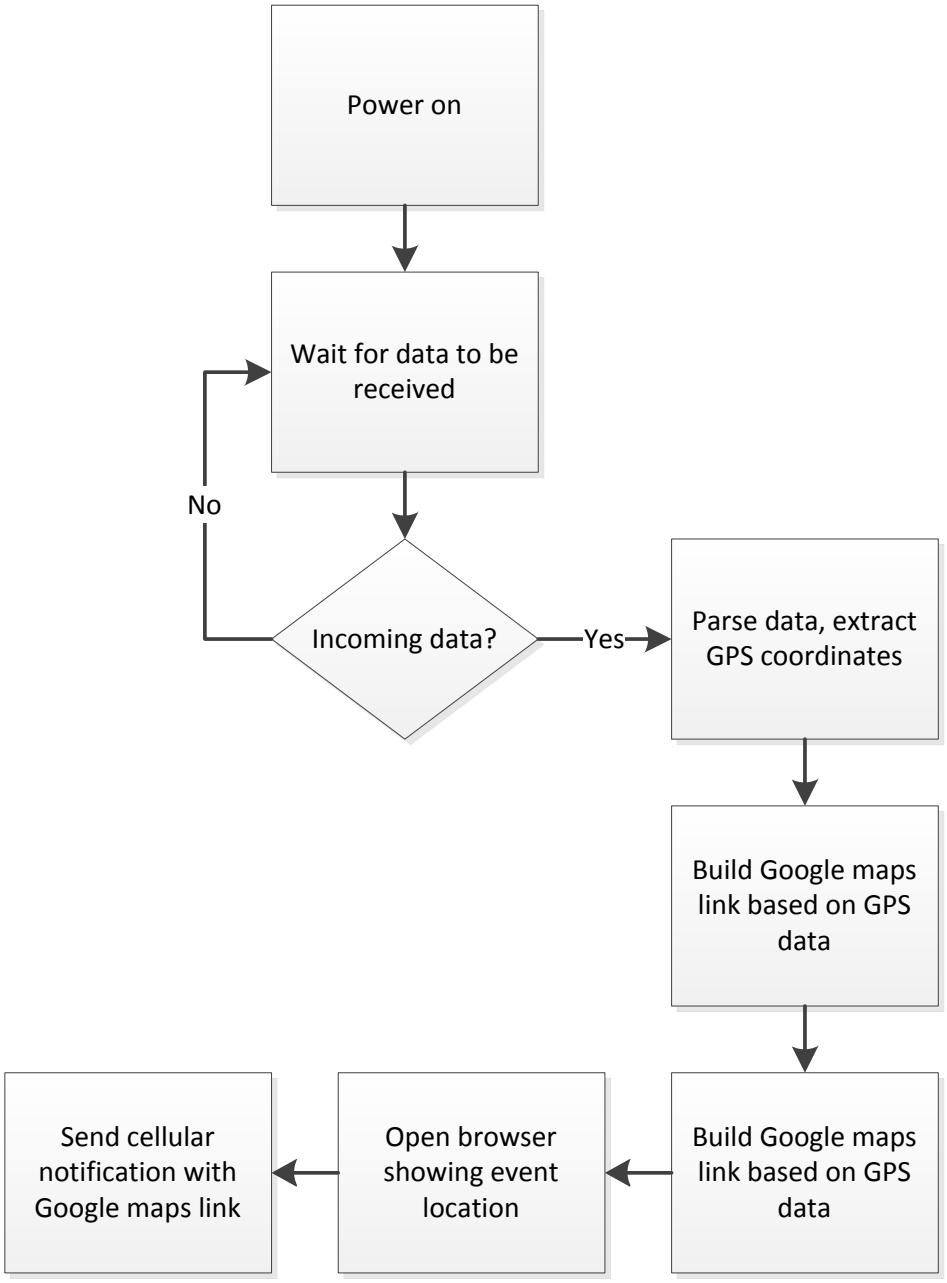

Figure 5: Prototype MATLAB software flow chart

Mechanical Description
Each device is 2.5in x 2.5in x 2in.

Specifications
The device can communicate wirelessly about 200yds with a 6V power supply.

This specification is theoretical, has yet to be verified.

Level of Effort
This project took about 4 weeks to define and create in full. $250 paid for all components and software (with the exception of the MATLAB license, which was a student license).

Cost Analysis
This section describes the cost of the components and supplies.

Tractor

Product	Description	Cost	Qty	Tot	Link
Arduino	controller	$ 35.00	1	$ 35.00	http://www.sparkfun.com/products/11021
Proto Shield	controller	$ 15.00	1	$ 15.00	http://www.sparkfun.com/products/7914
Xbee	wireless	$ 26.00	1	$ 26.00	http://www.sparkfun.com/products/10415
Xbee Explorer	dongle	$ 25.00	1	$ 25.00	http://www.sparkfun.com/products/8687
Gyro	LPY503AL	$ 30.00	1	$ 30.00	http://www.sparkfun.com/products/9424
Battery Holder	4x AA Barrel connector	$ 2.50	1	$ 2.50	http://www.sparkfun.com/products/9835
Total				**$ 133.50**	

End User

Product	Description	Cost	Qty	Tot	Link
Arduino	controller	$ 35.00	1	$ 35.00	http://www.sparkfun.com/products/11021
Proto Shield	controller	$ 15.00	1	$ 15.00	http://www.sparkfun.com/products/7914
Xbee	wireless	$ 26.00	1	$ 26.00	http://www.sparkfun.com/products/10415
Xbee Explorer	dongle	$ 25.00	1	$ 25.00	http://www.sparkfun.com/products/8687
Total				**$ 101.00**	

Grand Total	**$ 234.50**

Table 2: Prototype cost analysis

APPENDIX

PROTOTYPE SOFTWARE

ARDUINO CODE

```
float Aval1 = 0;  //define first analog read value
float Aval2 = 0;  //define second analog read value
float Vref = 0;
float maxV = 0;
float minV = 0;
float dVh = 0;
float dVl = 0;
float scaleFactor = 0; //scaling factor used in calibration
float zeroPoint = 233.55;  //define digital value that represents a stable device
float DtoA = 0.004883;  //define D to A conversion factor
float VtoDegPerSec = 0.033;  //define voltage to degree per second conversion factor
float DegPerSec = 0;  //define the angular velocity
float deg = 0;  //define initial degree (assume level initially)
float ddeg = 0;  //define incremental degree;
float dt = .004;            //define the sample period (0.0001s)
long ms1 = 0;
long ms2 = 0;
float numRead = 0;
float compensation = 34.56;
float AvalAvg;
float Aval_tmp;
float Aval;

void setup(){
 pinMode(15,INPUT);  //set z-axis analog input to INPUT
 pinMode(16,INPUT);  //3.3V reading, for calibration
 pinMode(17,INPUT);  //GND reading, for calibration
 pinMode(18,INPUT);  //VREF reading, for calibration
 Serial.begin(9600);  //start up the serial at 9600baud
 scaleFactor = 0.345;
}

void loop(){
 ms1 = millis();
 ms2 = ms1;
 while((ms2-ms1)<dt*1000){
  Aval_tmp = analogRead(1);
  if(Aval_tmp >=468){
   Aval_tmp = 468;
  }
  Aval = Aval + Aval_tmp;
  numRead++;
  ms2 = millis();
```

```
 }

 //calculate the deg per second
 AvalAvg = Aval/numRead;
 DegPerSec = ((((AvalAvg)-zeroPoint)*DtoA))/(VtoDegPerSec); //calculate angular velocity

 //update the current degree
 deg = deg + DegPerSec;


 if(deg > 4000){
  Serial.println("41.772112,-93.646774");
  while(1){} //stop, infinite loop
 }
 if(deg < -4000){
  Serial.println("41.772112,-93.646774");
  while(1){} //stop, infinite loop
 }
 Aval = 0;
 numRead = 0;
}
```

MATLAB CODE

Main Script

```
clear all
close all

comNum = input('Which COM port (e.g. 14)?: ','s');
comNum = ['COM',comNum];

s1 = serial(comNum,'BaudRate',9600);
fopen(s1);

disp('Proceeding to receive data...');

while(1)
   if (s1.BytesAvailable > 0)
      if (s1.TransferStatus == 'idle')
         pause(0.01);
         BA = s1.BytesAvailable;
         q = fread(s1, s1.BytesAvailable, 'char');
         q = q';
         fprintf('%s\n', char(q));
         break;
      end
   end
end
```

```
fclose(s1)
delete(s1)
clear s1

i = 1;
lat_val = [];
lon_val = [];

while(q(i) ~= 44 && i < 10)
   lat_val(i) = char(q(i));
   i = i + 1;
end
i = i+1;
while(i < 21)
   lon_val(i-10) = char(q(i));
   i = i + 1;
end

lat_val = char(lat_val);
lon_val = char(lon_val);
cell_notify(lat_val, lon_val)
```

Cellular Function

```
function cell_notify(lat, lon)
   latitude = lat;
   longitude = lon;

   myaddress = 'etalert.notify@gmail.com';
   mypassword = 'ETPhoneHome';

   setpref('Internet','E_mail',myaddress);
   setpref('Internet','SMTP_Server','smtp.gmail.com');
   setpref('Internet','SMTP_Username',myaddress);
   setpref('Internet','SMTP_Password',mypassword);

   props = java.lang.System.getProperties;
   props.setProperty('mail.smtp.auth','true');
   props.setProperty('mail.smtp.socketFactory.class','javax.net.ssl.SSLSocketFactory');
   props.setProperty('mail.smtp.socketFactory.port','465');

   prefix = ' https://maps.google.com/maps?q=';
   comma = ',';
   sentence = 'Potential rollover detected. Immediately assist the location ';
   web_location = [prefix,latitude,comma,longitude];
   body = [sentence,latitude,comma,longitude,web_location];

   sendmail('7123890381@vtext.com',body)
   web(web_location, '-browser')
end
```