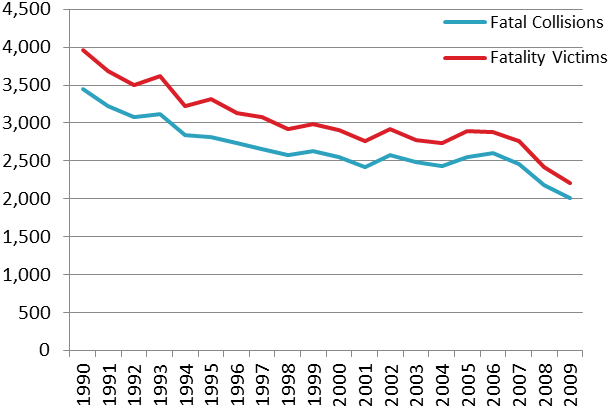
**Motivation:**

* Accident during lane changes are common due to driver’s not checking their Blind Spots
* Companies such as BMW, Volvo etc have implemented blind spot assist devices on their models, however, universal devices are not widely available
* Accident rates are decreasing as safety technology becomes more of a concern, but there’s room for improvement

*Provide a source the data (figure) below if you include this in your poster*



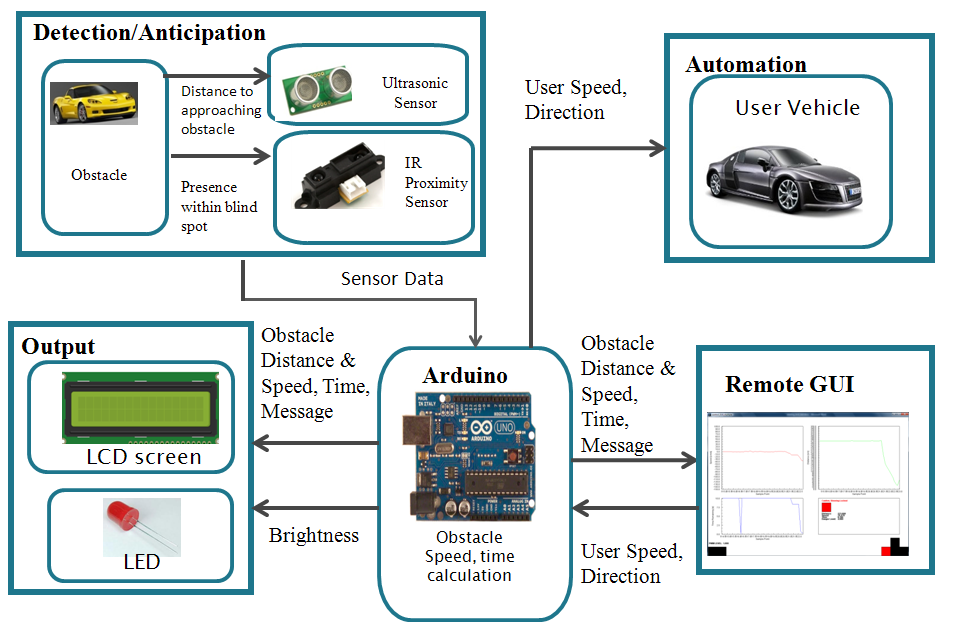
**Figure 5: Fatal Collisions and Fatality Victims (1990-2009)**

**Project Goal:**

* Assist in accident prevention through a device that will help users identify and anticipate obstacles in their vehicle’s blind spot
* Provide an additional level of accident prevention through the implementation of an automated steering lock feature based on relative danger levels

**System Level Overview:**

Perhaps provide a short description of the main flow of your design (maybe in bullet points, in addition to the figure below). Perhaps you should make it clear that the user vehicle has an arduino as well to provide the steering lock (automation) feature.

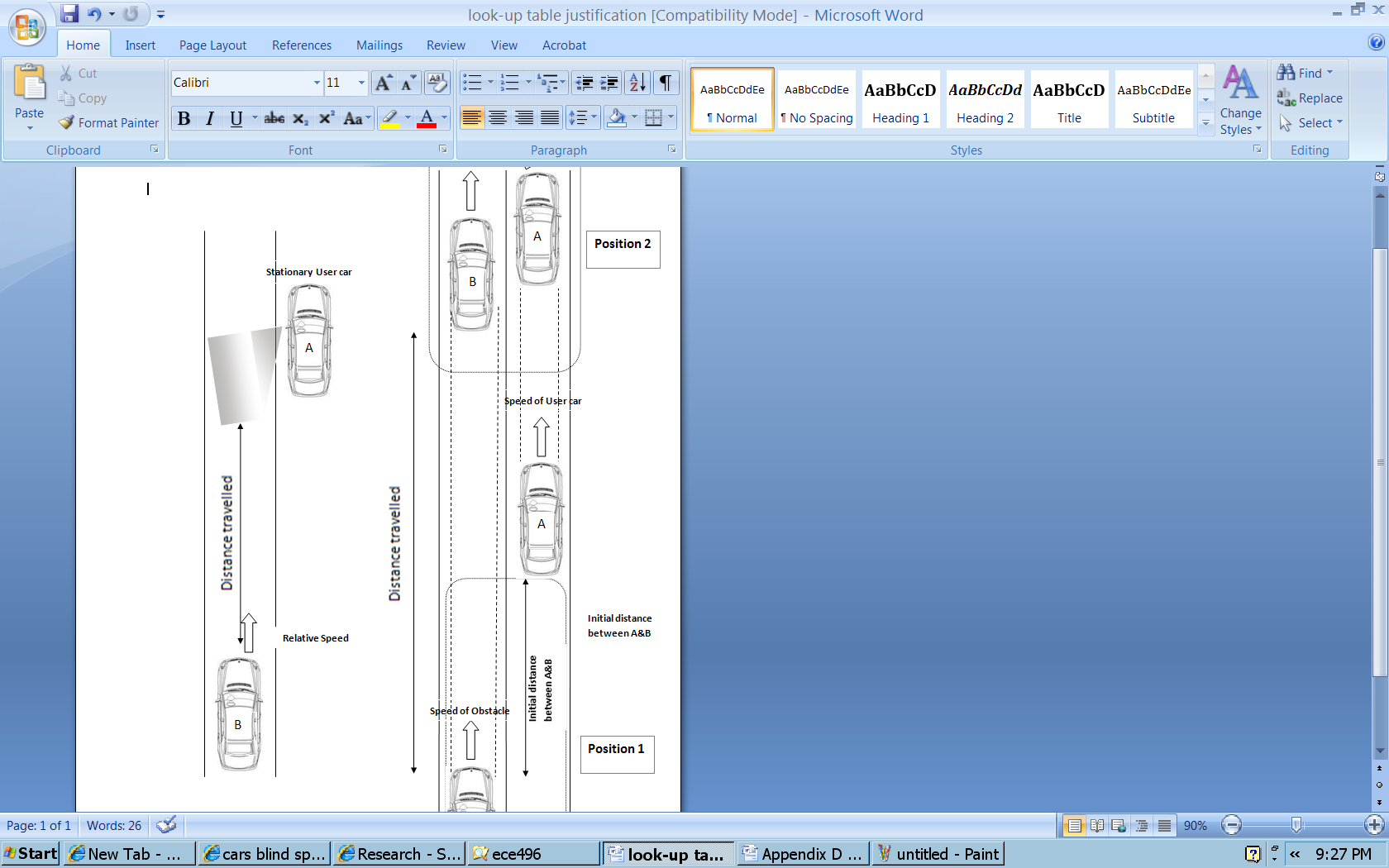
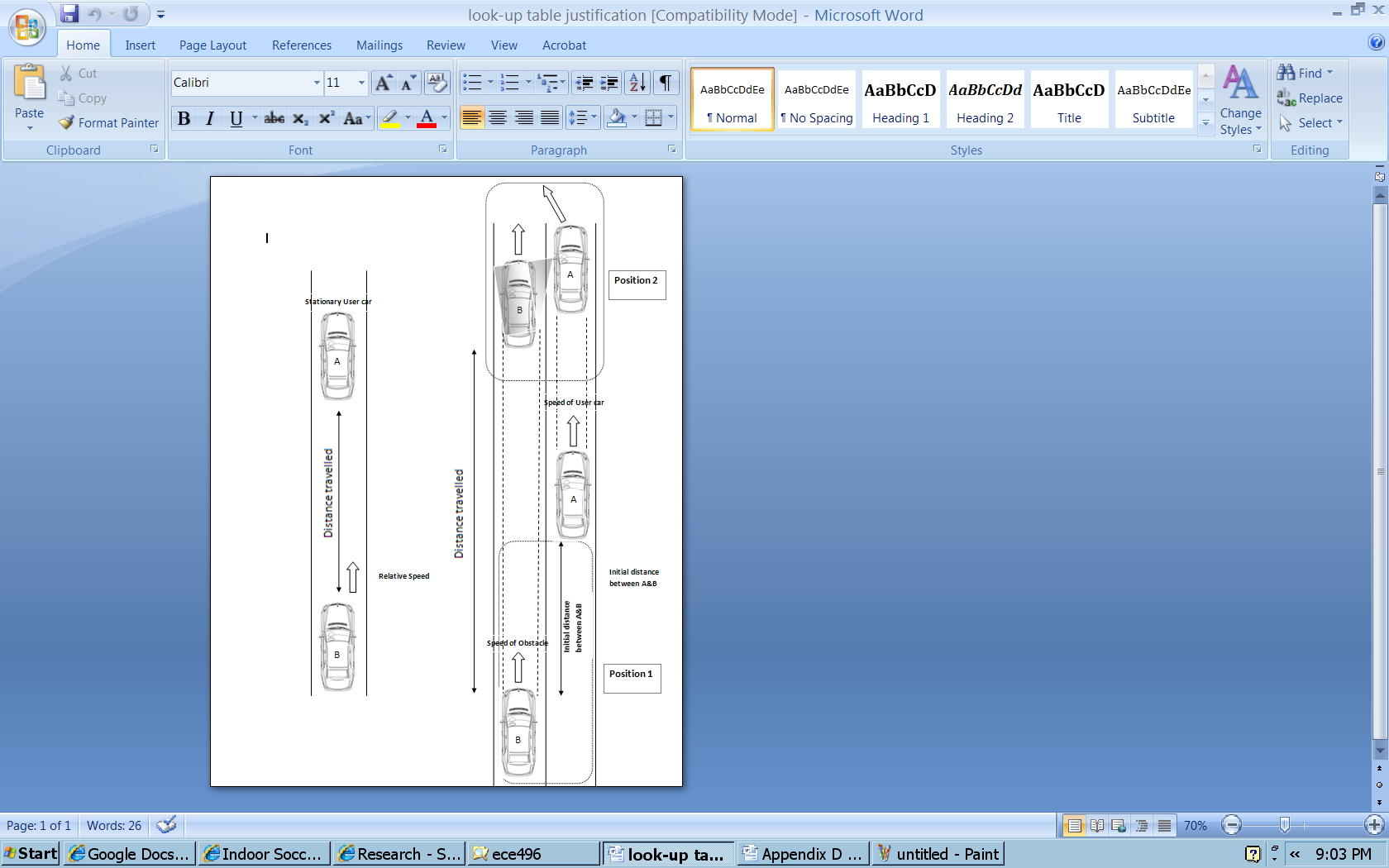


Process:

* Infrared (IR) sensor checks

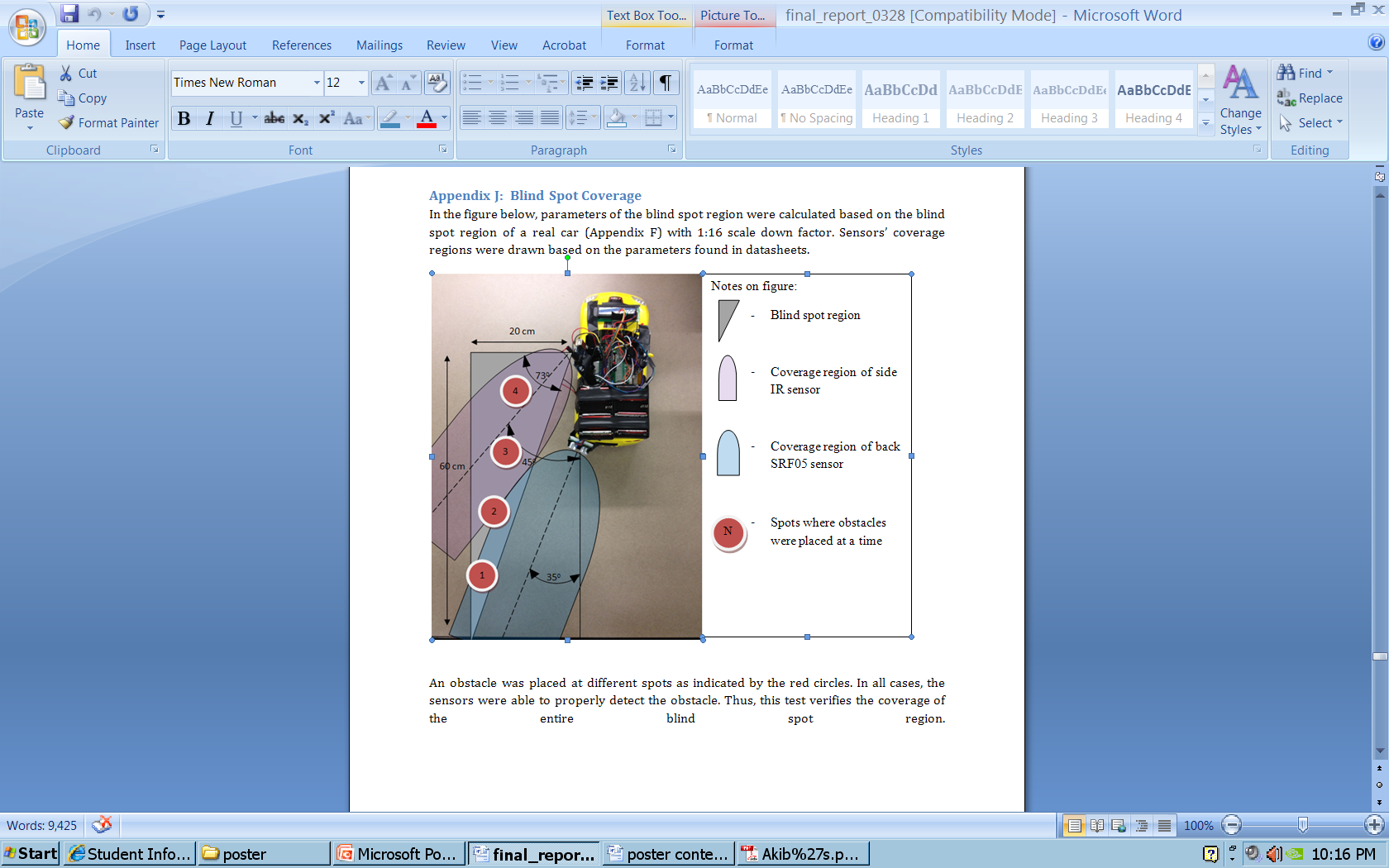
**Explanation of Anticipation/Detection:**

Again, perhaps a short description of the different cases would help make this figure more understandable.



**Sensor Coverage Areas**:

Very nice figure, which illustrates the complete coverage of the blind spot! Perhaps you should expand IR to be Infrared, and change SRF05 to ultrasonic (most people will not know what SRF05 is).



Coverage region of infrared sensor (for obstacle detection)

Coverage region of ultrasonic sensor (for obstacle anticipation)

Blind spot region

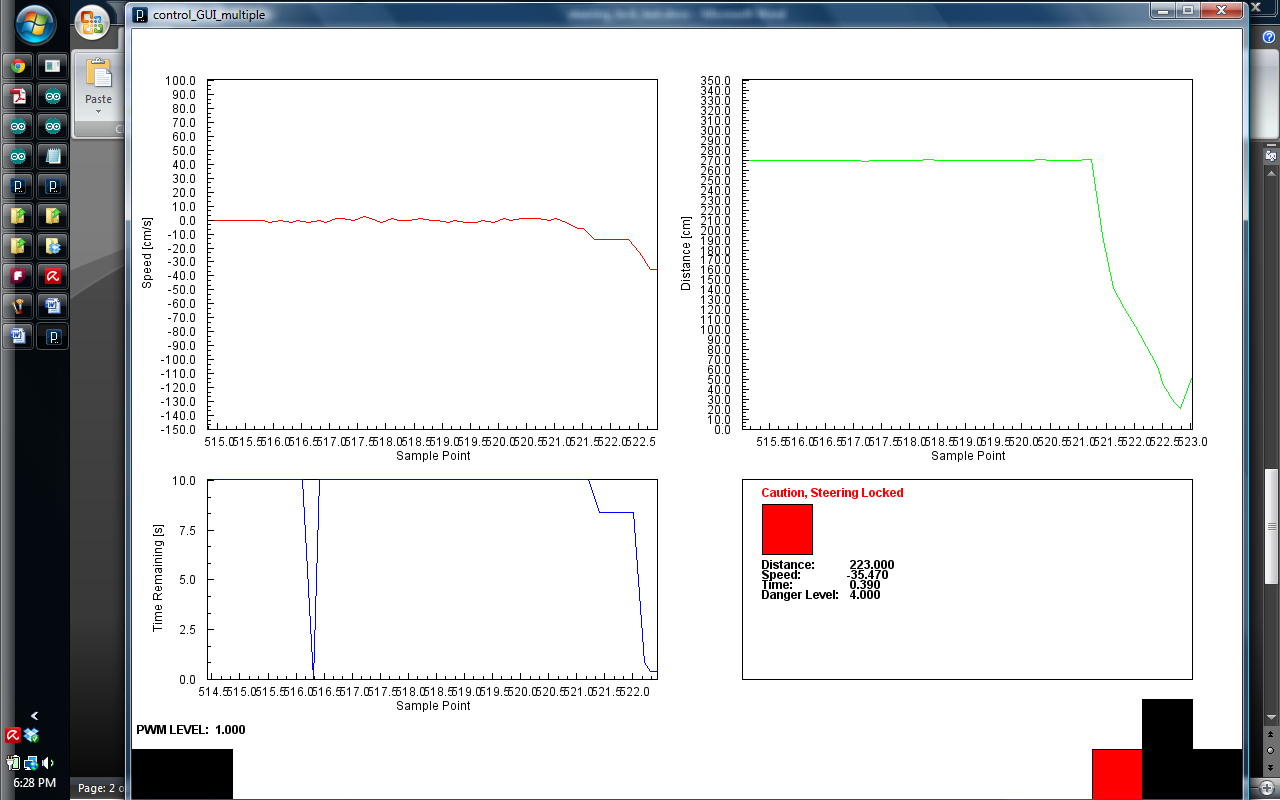
Spots where obstacles were placed

Graphical User Interface

Need a new picture for this, and it needs labelling

**Onboard Output**

**Wireless Output (GUI)**



LED

**Control Box Legend**

Key Not Pressed

Key Held

Key Disabled

**Increase/Decrease Max Speed**

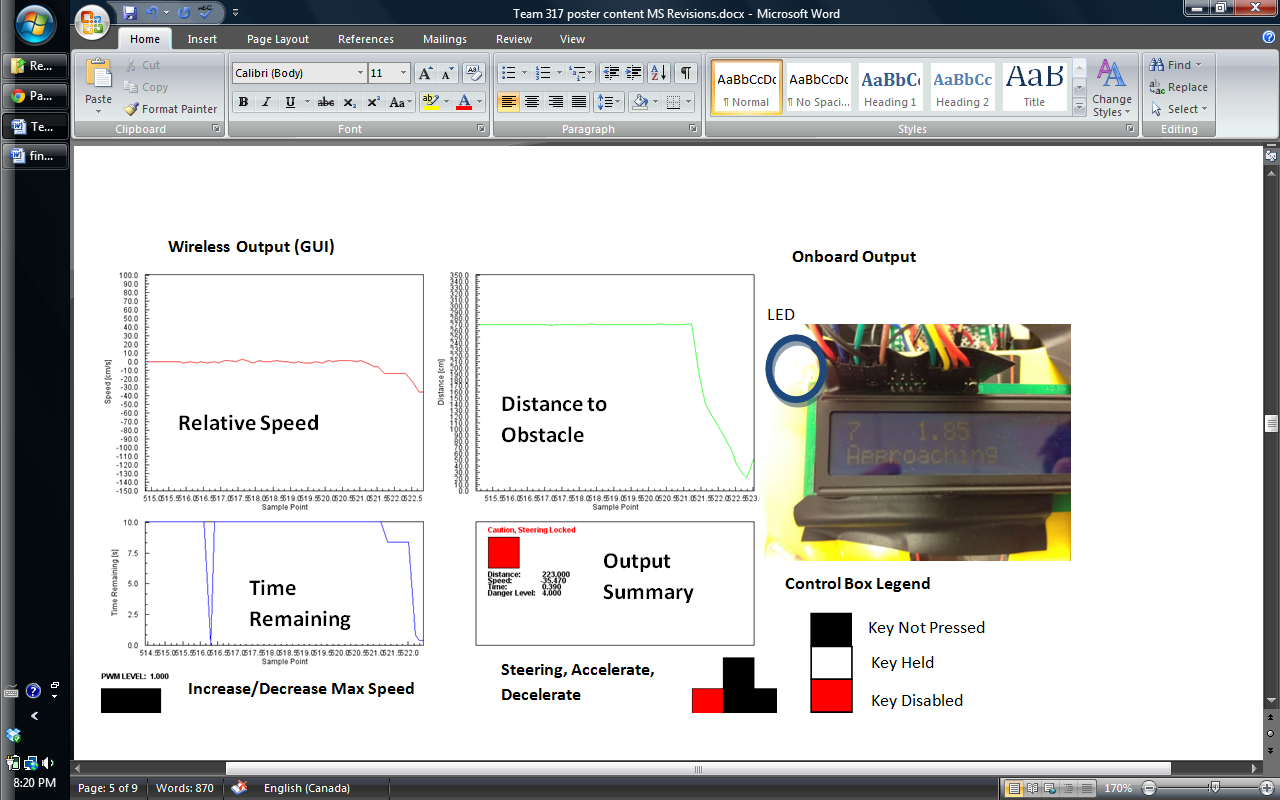
**Steering, Accelerate, Decelerate**

**Output Summary**

**Distance to Obstacle**

**Time Remaining**

**Relative Speed**



Yes, make sure you can read the labels on the figure…or describe what each box represents.

**Onboard Output:**

**Danger Levels and LCD Output Table: (Come up with a better name for this…so that the average person would understand what this table is, perhaps “Danger Levels”)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Situation** | **Sensor Involved** | **Time Left [s]** | **Danger Level** | **LCD Screen Output** | **LED Output** | **Steering Locked** |
| **No Danger** | Primary | t > 15 | 0 | Distance to obstacle, “No Danger” | Off | N |
| **Obstacle Approaching** | Primary | 11 < t < 15 | 1 | Distance to obstacle, relative speed, “Obstacle Approaching” | Brightness from 0% to 78%, depending on increasing distance | N |
| **Obstacle Approaching** | Primary | 6 < t < 11 | 2 | N |
| **Obstacle Approaching** | Primary | 3 < t < 6 | 3 | N |
| **Obstacle Very Near** | Primary | t < 3 | 4 | Distance to obstacle, relative speed, “Caution” | Fully On | N |
| **Obstacle at Blind Spot** | Secondary | t ≤ 0 | 5 | “Danger, Check Blind Spot” | Fully On | Y |

Summary of Test Results:

-Valik fill this in

**Provide a brief summary of these, in chart form perhaps (MS).**

**Onboard Software: (If you include this you should put it before your test results. This way you have a) motivated your project, b) stated the goal of the project, c) described the essential components of the design, d) summarized the test results which show your design solves the stated problem).**

Detection/Anticipation algorithm:

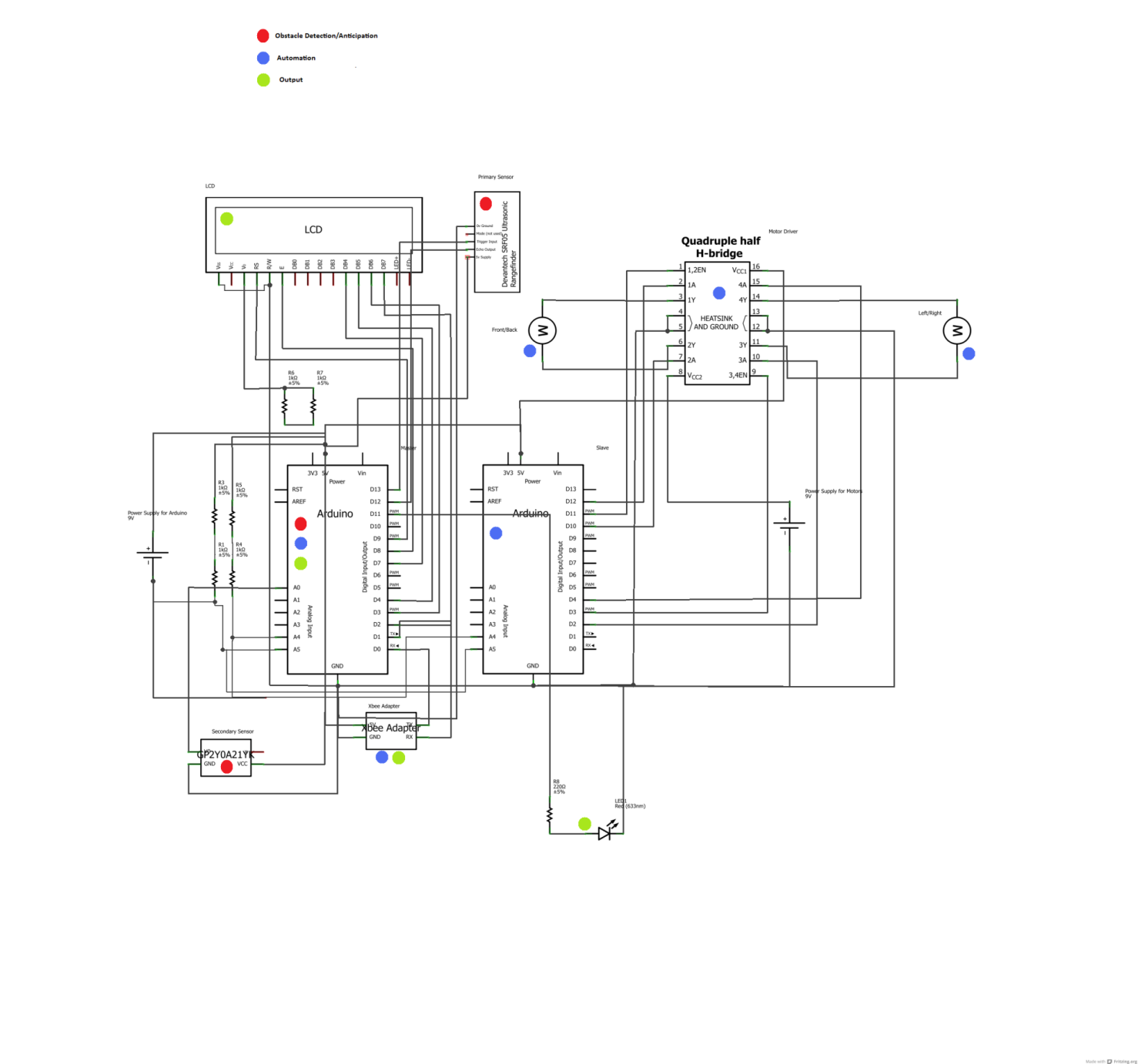
Make a flow chart?

* Check whether obstacle exists in the blind using the IR Sensor
* If not:
  + Use the ultrasonic sensor to check for approaching obstacles
  + If present, continuously gather distance values from the obstacles
    - Store the distance values in an array, and run a moving average to determine the average distance
    - Use consecutive distance values to calculate the change in distance, and use the internal clock to determine the change in time
    - Calculate instantaneous speed using the two previous values, run a similar moving average, and a sample and hold algorithm to obtain smoothed out relative speeds
    - Perform time remaining calulcations using:
      * t=d/v
    - Compare time remaining calculations against the look-up table, and provide corresponding feedback and/or lock steering
* Software Algorithm
  + Check whether obstacle exists in the blind Spot using the IR Sensor
    - If Present:
      * Warn user
      * Lock Steering
  + If the obstacle is not in the blind spot yet :
    - Use the ultrasonic sensor to check for approaching obstacles:
    - Continiously gather distance values from this obstacle, and strore the values in an array
    - Run a moving average on the array to determine distance to obstacle (D)
    - Use consecutive array elements to calculate change in distance (∆D)
    - Use the Arduino's internal clock to determine change in time (∆T)
    - Calculate instantaneous speeds using ∆V=∆D/∆T; Run a moving average to determin average relative speed (V)
    - Perform Time Remaining calculations (T) using: T = D/V
    - Warn user, and lock steering according to the Output Table

**Schematics:**

May be too technical, perhaps will need to take out

Yes, I would take these out (MS).



**List of components and Functions:**

May be too technical, perhaps will need to take out

Yes, I don’t think this level of detail is needed (MS)

|  |  |  |
| --- | --- | --- |
| **Component** | **Function** | **Module** |
| **Arduino (Master)** | * Drive the sensor * Perform distance, speed, and time calculations * Receive commands from the user * Transmit command to Arduino (slave), receive speed levels from Arduino (slave) * Drive output (LCD, LED, External) | Obstacle Detection/Anticipation  Output  Automation |
| **Arduino (Slave)** | * Operate the motor driver according Arduino’s (Master) commands * Track speed levels | Automation |
| **Motor Driver (SN754410)** | * Regulate the speed and direction of the two motors on the user vehicle (steering, front/back) according to Arduino’s (Slave) commands | Automation |
| **LED Light** | * Indicate danger levels with varying brightness | Output |
| **LCD Screen** | * Provide summarized information to the user | Output |
| **Ultrasonic (Primary) Sensor (SRF05)** | * Gather “time of flight” data to be used for distance values for obstacle anticipation | Obstacle Detection/Anticipation |
| **Infrared Proximity (Secondary) Sensor (GP2Y0A21YK)** | * Detect obstacles in the blind spot when it is out of the primary sensor’s range | Obstacle Detection/Anticipation |
| **Xbee Wireless Module (Series 1)** | * Gather wireless commands from user, and transmit sensor data wirelessly to be used in the GUI | Automation  Output |